IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 31 May 2023, accepted 6 August 2023, date of publication 22 August 2023, date of current version 5 September 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3307631

RESEARCH ARTICLE

Front-End Gateway System With Serial Communication Protocol Conversion and Edge Computing Platforms

GUO-MING SUNG, (Member, IEEE), LI-FEN TUNG, CHIH-JUNG HUANG, AND CHIH-PING YU
Department of Electrical Engineering, National Taipei University of Technology, Taipei 10608, Taiwan

Corresponding author: Guo-Ming Sung (gmsung@ntut.edu.tw)

This work was supported by the National Science and Technology Council, Taiwan, under Contract MOST 111-2622-E-027-012.

ABSTRACT This study presents a front-end gateway system with asynchronous and synchronous serial communication protocol conversion between a universal asynchronous receiver/transmitter (UART) protocol and an inter-integrated circuit (I^2C) protocol. The valid data and sensing data can be integrated using the I^2C protocol and sent to an edge computing platform for automated data analysis. Our system reduces the data processing time and the number of buses. The system has an edge computing platform that handles simple linear regression, base conversion, a neural network, and a text database and communicates with multiple peripheral devices. The I^2C master and slave are constructed on the edge computing platform and implement arbitration by using the carrier-sense multiple access with collision avoidance protocol to prevent data collision. According to the results obtained from the Signal Tap logic analyzer in experiments conducted using a field-programmable gate array board, a completed 330-bit UART packet requires 755.2 μs to be received, and the throughput is 436.97 kbps. By contrast, a 90-bit I^2C packet requires 184.6 μs to be received, and the throughput is 487.54 kbps. The front-end gateway sends integrated packets by using the I^2C protocol, and the operating freuency (serial clock) of the I^2C slave can reach up to 3.6 MHz bidirectionally. An integrated 153-bit packet requires 42.96 μs to be received by the edge computing platform, and the throughput is 3.5614 Mbps, which is approximately 8.15 times higher than that of the UART packet. We also fabricated a front-end gateway ASIC by using the TSMC 90-nm 1P9M CMOS process.

INDEX TERMS Universal asynchronous receiver/transmitter (UART), inter-integrated circuit (I^2C), asynchronous communication, edge computing, asynchronous transfer mode, ZigBee, Internet of Things, digital integrated circuits, field programmable gate arrays (FPGA), application specific integrated circuits (ASIC).

I. INTRODUCTION

A front-end gateway system is proposed in this paper; this system has a multimaster structure that prevents transmitted data from interfering with each other by using a wired-AND configuration and the carrier-sense multiple access with collision avoidance (CSMA/CA) protocol [1]. Specifically, CSMA/CA is a basic medium-access control (MAC) protocol specified in the popular IEEE 802.11 standard. A key feature of CSMA/CA is the use of in-band control frames for handling hidden terminals and reducing the cost of data interference. In general, the gateway used to collect data in a sensor network must be highly flexible for it to be compliant with the latest communication standards [2]. All received signals can be transmitted with different features in response to the signal propagation conditions. Collision prevention and suitable transmission must be ensured in a gateway system. This study's system ensures collision avoidance by using sketchy frames to deliver control information instead of bit-based control frames, as is the case in traditional MAC protocols. A sketchy frame encodes control information despite containing no meaningful bits; it does so

The associate editor coordinating the review of this manuscript and approving it for publication was Giovanni Merlino.

through its length, which is closely related to its transmission time [3].

The present study also formulated a new carrier-sense multiple access with collision resolution protocol. This protocol is used for collision resolution; specifically, when a collision is detected, a jamming signal that stops the transmissions of other stations is released, and the data are retransmitted without a backoff procedure. This step ensures the success of retransmission after the detection of a collision, which is a key feature of the proposed MAC protocol [4].

Edge computing is implemented using container-based virtualization for data-driven analysis and prediction [5]; such computing is highly useful for the platform controllers and process modules of Internet of Things (IoT) platforms and can be used to treat an edge processing workflow as a data flow model in accordance with user requirements [6]. In general, an edge computing platform has an inter-integrated circuit (I^2C) master and slave that are combined for serial communication. The protocol conversion unit (PCU) enables a serial peripheral interface master device (sender) to communicate (specifically, send data) with an I^2C slave device (receiver).

A previous study fabricated an edge computing platform using Xilinx ISE 14.7 and simulated it using ISIM software [7]. The I^2C bus specification of [8] shows that the data on the I^2C-bus can be transferred at rates of 3.4 Mbps in the high-speed mode and the clock frequency of serial clock (SCL) can reach 3.4 MHz for the I^2C-bus devices. In general, I^2Cs have a half-diplex and two-wire interface and are popular because of their simplicity; they enable communication between devices of different speeds without data loss. For example, an I^2C can have a field-programmable gate array (FPGA) board acting as the master (faster device) and an MEMS motion sensor (slow device) acting as the slave. Because of their wealth of features, I^2Cs will continue to be highly popular as serial interfaces for connecting integrated circuits on a board [9]. In this study, our front-end gateway was evaluated on an FPGA development board with a Cyclone V chip; this board allowed us to examine simulated waveforms by using the Signal Tap logic analyzer program [10].

In this study, we fabricated a chip for the proposed front-end gateway system with asynchronous and synchronous serial communication protocols using TSMC 90-nm CMOS technology. The proposed IoT network not only receives and stores sensing data using an FPGA board but also uses edge computing to output predictions easily on the basis of sensing data. The gateway system application-specific integrated circuit (ASIC) accelerates integrated packet transmission with arbitration to prevent data collision by using the CSMA/CA protocol. The rest of this paper is organized as follows. Section II describes the system architecture of the proposed front-end gateway ASIC with serial communication protocol conversion, and Section III details the edge computing platform. Section IV describes the evaluation experiments, in which an FPGA was used. Finally, Section V provides the conclusions of this study.

II. SYSTEM ARCHITECTURE OF THE PROPOSED FRONT-END GATEWAY ASIC

Fig. 1 illustrates the system architecture of the proposed IoT network with a front-end-gateway ASIC, which runs on an I^2C-based serial communication protocol [8]. The system architecture comprises front-end, medium-end, and back-end parts. The front-end part is used for data processing, specifically data receiving, packet processing, checksum verification, and protocol conversion. The medium-end part is used for edge computing and to run online simple linear regression and a neural network (NN) and to store data. The back-end part is used for data monitoring and storage and to display the sensing data and store data offline in electrically erasable programmable read-only memory (EEPROM).

Fig. 2 illustrates the flow of sensing data in the proposed front-end gateway system. Sensing data are communicated using two methods. The first method involves wireless communication with ZigBee, where the system is connected to a ZigBee module and generates universal asynchronous receiver/transmitter (UART) packets after a ZigBee receiver is connected at the front end [11], [12]. The second method involves wired communication with an onboard sensor, which is connected to I^2C master A. The front-end gateway sends periodic fetch commands to obtain sensing data from the onboard sensor and to process the obtained packets by using I^2C master A. The data received from I^2C master A are integrated with UART packets and sent to the edge computing platform only after passing checksum and cyclic redundancy check (CRC) verification. The aforementioned integrated packets are received by the I^2C slave at the edge computing platform. After the received packets have been verified, the real-time linear regression and an NN are completed on

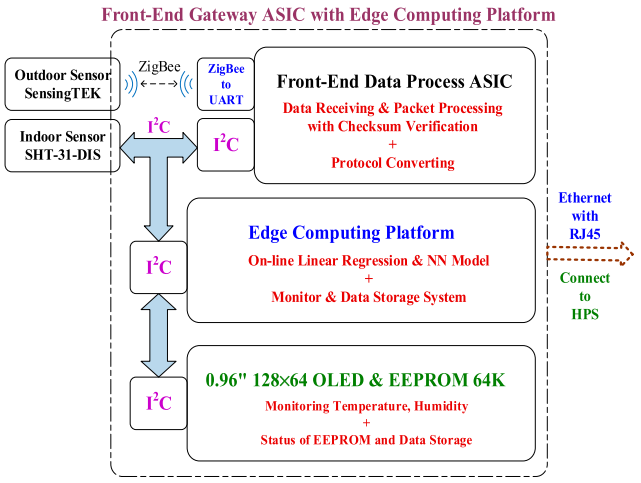

FIGURE 1. Architecture of the proposed IoT network with a front-end gateway ASIC.

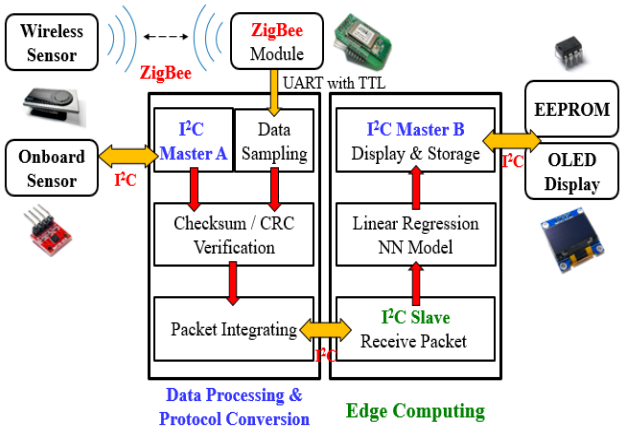

FIGURE 2. Flow of sensing data in the proposed front-end gateway ASIC.

hardware for data analysis and prediction. I^2C master B displays the predicted data on an organic LED display.

A. FINITE-STATE MACHINE OF I^2C MASTER A

The I^2C communication protocol is used to transmit data from an onboard sensor [13]. The address of the I^2C slave, which is one of several sensing devices, is set as "0×44." To enable bidirectional communication between the onboard sensor and the edge computing platform, a finite-state machine (FSM) is used in the front-end gateway ASIC of the I^2C master A. Fig. 3 illustrates the FSM of the I^2C master for onboard sensor. In the proposed FSM, the operating mode and state flow of the ASIC are formulated to ensure that the designed functions can be completed in a sequential manner. The SCL frequency is set at 0.5 MHz on the basis of the device address, acquisition command specifications, and data parameters in the datasheet of onboard sensor. The gateway ASIC has six modes of operation that are switched using a 2-bit switch (SW). The first and second bits are the most significant

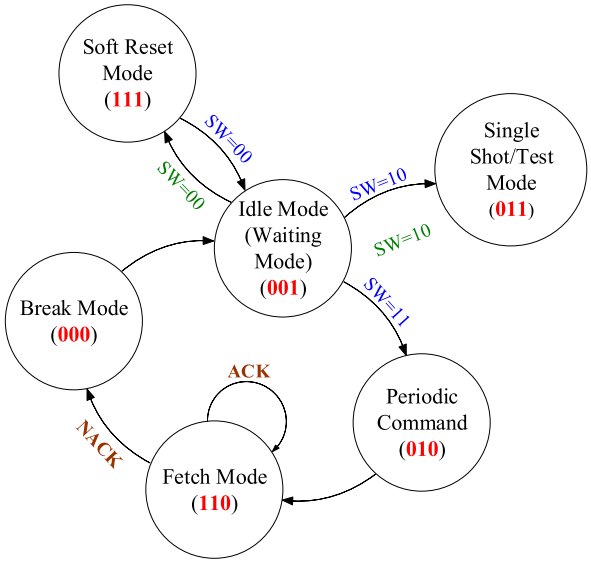

FIGURE 3. FSM of the I^2C master in the onboard sensor.

bit (MSB) and least significant bit (LSM), respectively. The six modes of the ASIC are as follows:

1) Idle mode/waiting mode: In this mode, the system is idle and waits for the user to trigger the gateway by using an externally mounted hardware button. The state flow is determined using 2-bit switches.
2) Soft reset mode/measurement mode: In this mode, the calibration value is automatically loaded, and the sensor status register is updated. If the status registers as abnormal, the user can switch to the idle mode by pushing the externally mounted hardware button to calibrate the value of the sensor.
3) Single shot mode/verification mode: In this mode, the measurement command is sent first, and the system then immediately switches to a receiving state to capture a single packet of data from the sensor. After receiving the packet, the front-end gateway performs a CRC, thus reducing the time required for verification.
4) Periodic command/working mode: In this mode, the ASIC operates on the basis of periodic measurements performed without interruption. A periodic measurement command must be sent to the sensor in the I^2C network for periodic measurement to be executed. The I^2C master then switches to the fetch mode and awaits the fetch command.
5) Fetch mode: In this mode, a fetch command is sent 2 s after a periodic command is issued. If the fetch command is sent too quickly, the sensor responds with a negative acknowledgment in the acknowledgment (ACK) stage to invalidate the periodic command because the sensing data are not ready in the register. In response, the sensor automatically switches to the break mode. If the fetch command is sent successfully, the sensor returns an ACK in the ACK stage.
6) Break mode: This mode is activated when a running procedure stops because of an error in the code or the presence of deliberate interference in the data transfer process. In this mode, the sensor automatically switches to the idle mode.

B. OPERATION FLOW OF THE FRONT-END GATEWAY SYSTEM

An accurate wireless sensor that measures humidity and temperature (SHT31-DIS) was developed by SensingTek [14]. Our front-end gateway is used to connect the sensing device and implement checksum and CRC algorithms. The front-end gateway is designed to guarantee the validity of the transmitted packet, and the operation flow of this gateway is illustrated in Fig. 4. Specifically, the outdoor and indoor sensing data must pass checksum and CRC verification.

C. CHECKSUM VERIFICATION

The operation flow of checksum verification is illustrated in Fig. 5. A byte stuffing flag is used to detect and store the next piece of data; these data are subject to XOR calculations

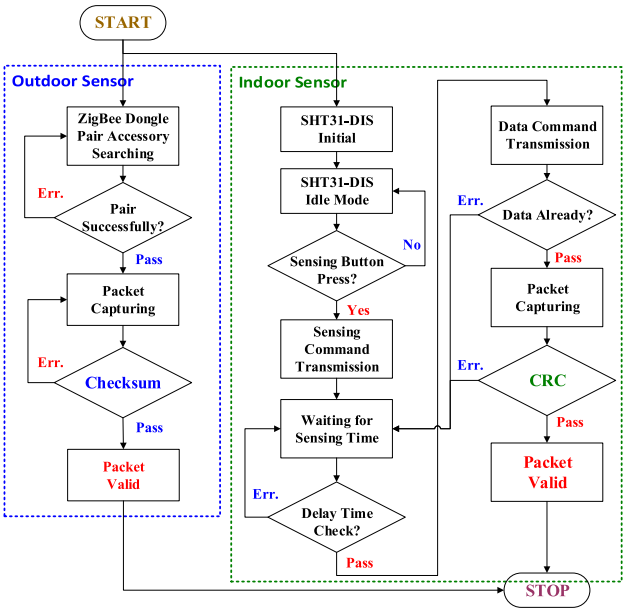

FIGURE 4. Operation flow of the proposed front-end gateway ASIC.

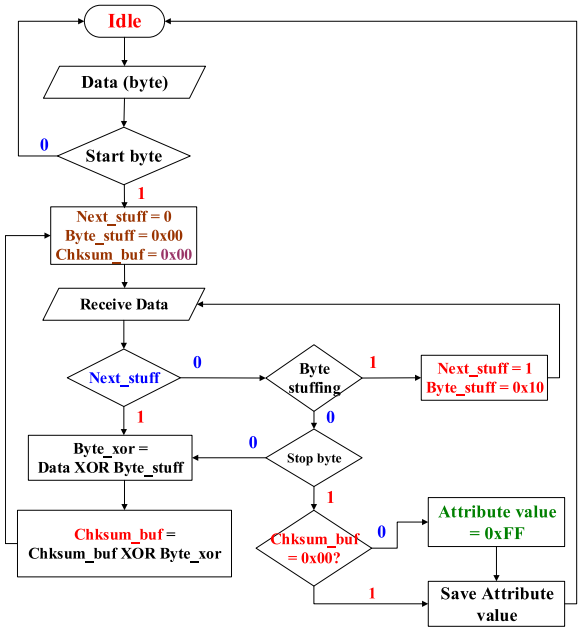

FIGURE 5. Operation flow of checksum verification.

with the byte stuffing value (0 × 10) in Chksum_buf. After the stop byte (1) is received, the checksum system verifies the checksum value in Chksum_buf. The data are error-free if the checksum value is 0 × 00. The attribute value of these data is then set to 0xFF, which denotes error-free data. Moreover, during the idle state, the data line is maintained in a high-potential state to ensure that erroneous data are not captured in the packet. The checksum value of erroneous data is 0×00, which is also their attribute value and the initial value of Chksum_buf.

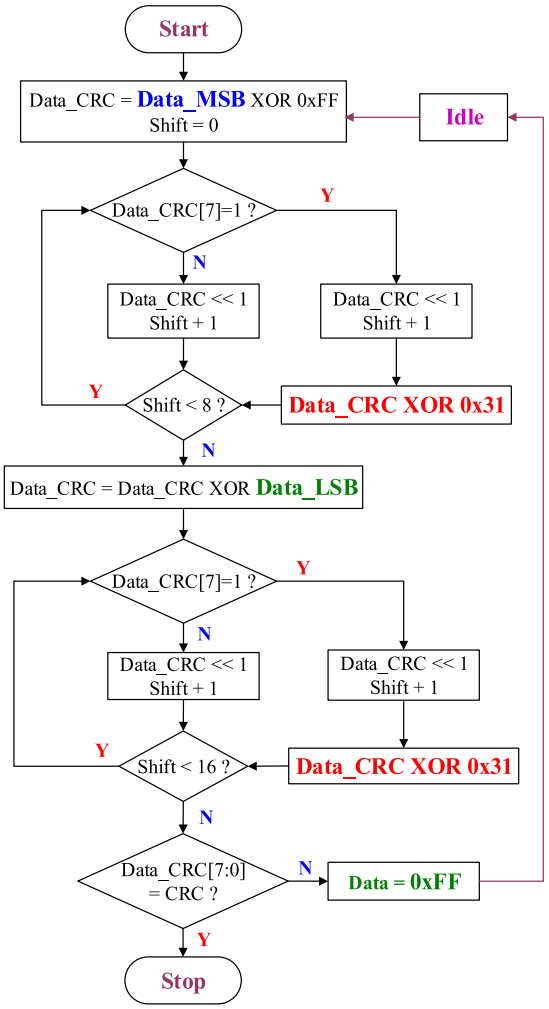

FIGURE 6. Operation flow of CRC verification.

D. CRC VERIFICATION

The operation flow of CRC verification is illustrated in Fig. 6. Our proposed design is based on the CRC-8 rule, which is used to generate the CRC value according to the ITU-IEEE specification [15]. An 8-bit CRC checksum is generated by the CRC algorithm after the sensing data are received. The CRC-8 properties of the CRC algorithm are listed in Table 1. The CRC data are encoded as two transmitted bytes that are used to calculate the checksum value: Data_MSB and Data_LSB. First, the XOR operation is completed using Data_MSB and an initial value of 0xFF. Subsequently, if the highest bit of the CRC checksum Data_CRC [7] is high (1), the left-shift operation (≪) is executed to obtain the remainder of the binary division and to perform the XOR operation with the polynomial value of 0 × 31. If the Data_CRC [7] value is low (0), the left-shift operation (≪) is used to execute binary division. The lower byte of the sensing data (Data_LSB) is subjected to a similar series of operations. Thereafter, the final CRC checksum is determined.

TABLE 1. CRC-8 property of the CRC algorithm.

Property	Quantity
rule	CRC-8
width	8 bits
protected data	Read and/or write data
polynomial	0x31 ($x^8+x^5+x^4+1$)
initialization	0xFF
reflect input	False
reflect output	False
final XOR	0x00

III. EDGE COMPUTING PLATFORM

The present study adopted an edge computing platform with a costructured architecture and an I^{2}C protocol. The I^{2}C slave receives the sensing data, analyzes the integrated packet, and determines the trend of the received packets through on-time linear regression. The calculated data are then sent to an NN for prediction. Sensing and prediction data packets are transmitted using the I^{2}C protocol. The I^{2}C slave is used to receive the sensing data with the front-end gateway system, and the master is used to control the back-end peripheral display and the storage devices [16], [17]. Fig. 7 illustrates the operation flow of the edge computing platform.

A. I^{2}C SLAVE ARCHITECTURE

A specially designed I^{2}C slave is used in the edge computing platform to receive the integrated packet; its operation is as follows (Fig. 8). First, start–stop logic is used to detect the initiation and completion of the operation of the I^{2}C slave. The slave address comparator is then used to verify whether a sensing signal has been received. Subsequently, the ACK logic of the I^{2}C master is used to control the control signal SDA_OEN [18].

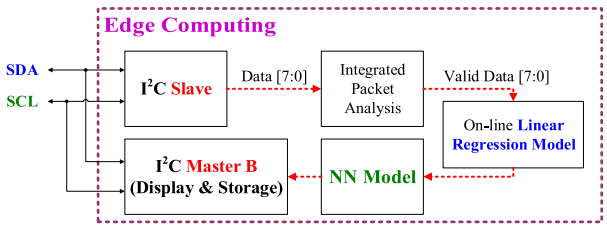

FIGURE 7. Data flow in the edge computing platform.

The CSMA/CA protocol is used to prevent collision in the I^{2}C bus. The protocol monitors the serial data (SDA) and SCL signals and the START and STOP signals to determine whether sensing data have been fully transmitted by other master devices. The system compares the signal SCL_OEN with the signal SCL to decide whether the present channel (bus) is busy, which is indicated by workingflag_CSMA.

Fig. 9 presents simulation results for CSMA/CA conditional detection, gateway command execution, and edge platform initialization. In the first period of workingflag_CSMA

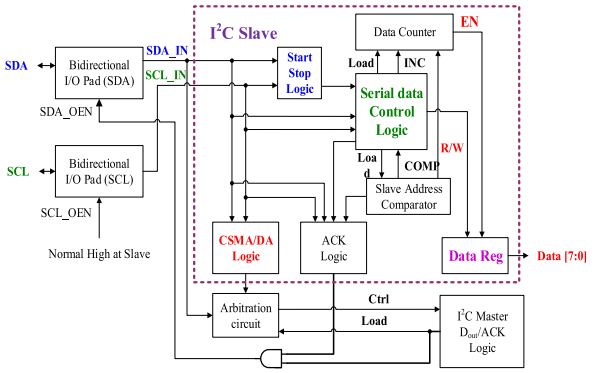

FIGURE 8. Operation flow of the *I^{2}C* slave architecture.

(denoted by ①), the gateway command of the test mode is executed; consequently, a single packet is received to verify that the sensor is operational. Subsequently, the edge platform initialization of the EEPROM is initiated in the second period of workingflag_CSMA (denoted by ②).

B. SIMPLE LINEAR REGRESSION

An initial linear model of calculated temperature y_t and data ($\Delta T = 20$ s) x_t is established in $y_t = \beta_1 x_t + \beta_0$. Least squares regression can be used to determine the optimal values of β_1 and β_0. Specifically, least squares regression yields the sum of the square deviation (SS_X) and the sum of cross product SS_{XY} for X and Y when evaluating the predicted values b_0 and b_1 of β_0 and β_1, respectively [19]. Table 2 presents an example where seven historical data points on temperature, which are measured by the sensor, are used to predict future temperature.

$$\overline{X} = \sum_{i=1}^{n} x_t(i) \Big/ n = 4 \tag{1}$$

$$\overline{Y} = \sum_{i=1}^{n} y_t(i) \Big/ n = 29 \tag{2}$$

$$SS_X = \sum_{i=1}^{7} [x_t\,(i) - 4]^2 = 28 \tag{3}$$

$$SS_{XY} = \sum_{i=1}^{7} [x_t\,(i) - 4] \times [y_t\,(i) - 29] = 0.56 \tag{4}$$

Correspondingly, $b_1 = SS_{XY}/SS_X = 0.56/28 = 0.02$ and $b_0 = 29 - 0.02 \times 4 = 28.92$, and the linear regression model is $y_t = 0.02\ x_t + 28.92$. Fig. 10 presents the fitted curve and data points. The mean squared error (MSE) of the linear regression model is approximately 0.5606.

C. NEURAL NETWORK (NN)

A nonlinear autoregressive time-series model was established in MATLAB NN Toolbox software for training and prediction. Fig. 11 illustrates the established NN architecture. During training, the weight of the input layer is fed to each

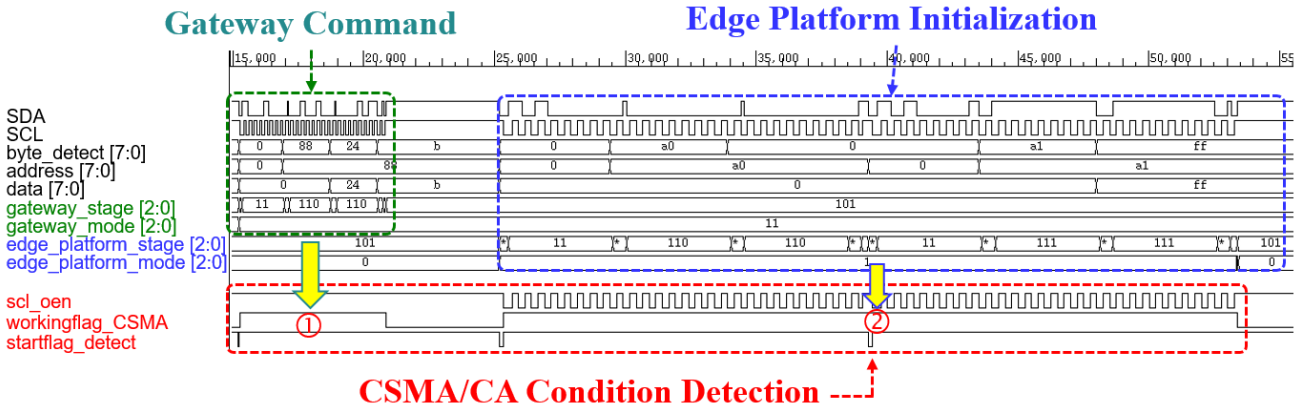

FIGURE 9. Simulated results for CSMA/CA conditional detection, gateway command execution, and edge platform initialization.

TABLE 2. Seven sensed temperature values.

data (x_t) (ΔT = 20 seconds)	temperatures (y_t)
1st Data	28.97
2nd Data	29.02
3rd Data	28.87
4th Data	28.98
5th Data	29.04
6th Data	29.02
7th Data	29.10

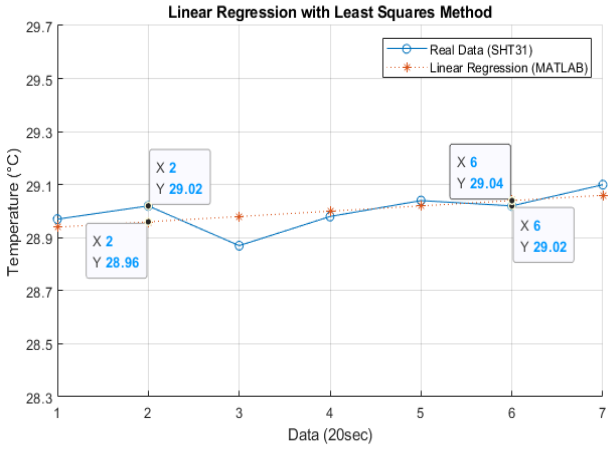

FIGURE 10. Linear regression on seven temperature data points.

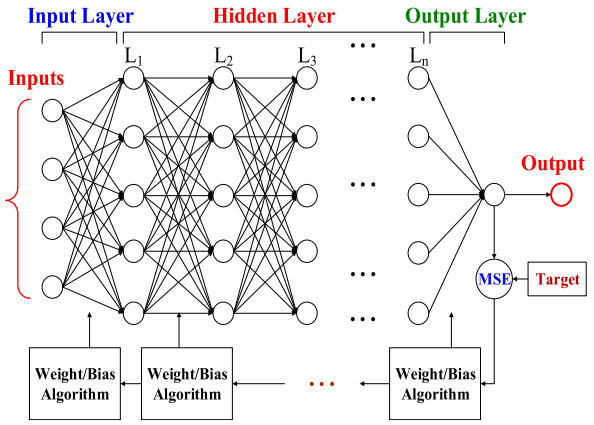

FIGURE 11. Architecture of the established NN.

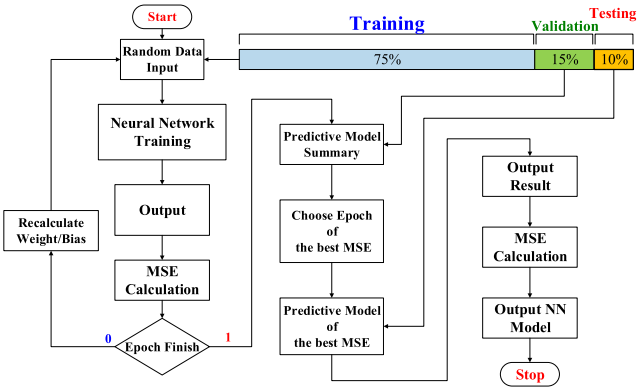

FIGURE 12. Training flowchart of the established NN.

neuron of the hidden layer and then activated using the activation function. After the output value is obtained, the weight is continually calculated and passed to the subsequent hidden layer. Finally, the final value is output through the output layer. To evaluate the output, the MSE is calculated as the difference between the final output value and the target value, which serves as feedback for weight correction. The MSE is defined in [20]. Ideally, the output value of the network approaches the target output value as training progresses [21].

Fig. 12 illustrates the training flowchart for the established NN; 75%, 15%, and 10% of the data are used for training, validation, and testing, respectively. During training, the MSE is calculated in each epoch, and the network parameters are adjusted. Subsequently, the output results of all epochs are recalculated using the validation set. The MSE for the test set in each epoch is used to obtain the best prediction model (i.e., the model with the best generalization ability).

IV. EVALUATION EXPERIMENTS

An experimental platform was used to evaluate the proposed front-end gateway system (Fig. 13). In this platform, two

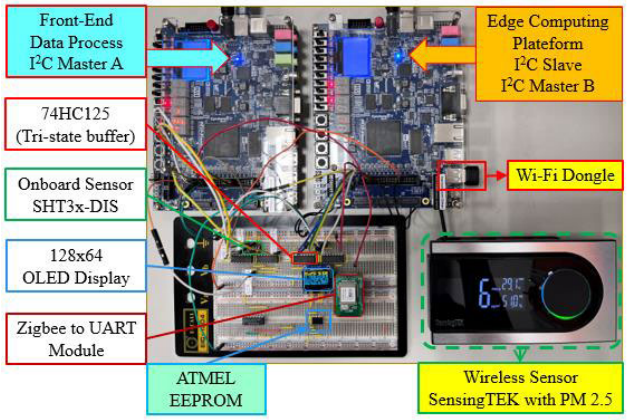

FIGURE 13. Verification environment of the proposed front-end gateway system.

DE 10-Standard development boards (i.e., FPGA boards) are used for the validation of bidirectional communication [22]. The wireless sensor transmits sensing data on the basis of the ZigBee wireless protocol. The sensing data are then converted according to the UART protocol by using a TTL circuit and fed to the FPGA board through a general purpose input/output (GPIO) interface. The packet of the onboard sensor is processed by the I^2C master. The edge computing platform handles OLED initialization commands and the character library. This platform is used to detect device abnormalities and to display the received data. The EEPROM is used to store the sensing data offline.

A. BIDIRECTIONAL I/O PORT VERIFICATION

The FPGA board of the proposed front-end gateway system does not have a bidirectional I/O port in its GPIO interface, which makes verifying the output of the bidirectional I/O port difficult. Specifically, the sensing data cannot be transmitted bidirectionally with a single wire. Therefore, only one signal I/O port was selected to be an input pin or output pin in the pin planner stage. Consequently, a tri-state buffer was required for bidirectional transmission to be verified with a single wire. Thus, we used a 74HC125 chip, which provides four independent buffer gates with three-stage outputs, in the bidirectional I/O port with a single wire (Fig. 14).

Fig. 15 illustrates a measured UART packet captured by the Signal Tap logic analyzer on the FPGA board [22]. The clock frequency of the sensing device was 5 MHz, the baud rate was 0.5 MHz, and the total number of sampling points was 3776 points. A completed 330-bit packet was received in approximately 755.2 μs, and the throughput was approximately 436.97 kbps. The start byte was 0×01 in hexadecimal notation (h; Fig. 15). The LSB and MSB were transmitted first and last, respectively. The output address, attribute, and value of the sensor were 5692h, 0012h, and 0002h, respectively. Finally, the checksum was verified as 0xB7h after the execution of the XOR operation with byte stuffing, which indicates that the next byte had to have a value of 0×10h after the XOR operation when the receiving byte was 0×02h.

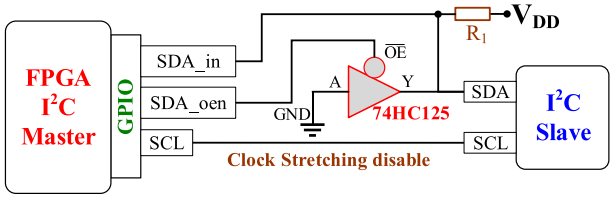

FIGURE 14. Bidirectional I/O port with a single wire that was implemented using the 74HC125 device.

Fig. 16 presents the plot of a measured I^2C packet captured by the Signal Tap logic analyzer. The operating frequency of SCL was approximately 0.5 MHz. A 90-bit packet was transmitted at approximately 184.6 μs, and the throughput was approximately 487.54 kbps. The front-end gateway sent the integrated packets by using the I^2C protocol. The operating freuency (SCL) of the I^2C slave could reach up to 3.6 MHz bidirectionally. The write and read bits of the slave address were 0×88h and 0×89h, respectively. Given this information, the system could find two received binary codes in the received signal (sda_in): "1000_1000" and "1000_1001" for the write and read bits, respectively. In the same expression, the hexadecimal code (0×000E) of the fetch command was expressed as the binary code "1110_0000_0000_0000" in the signal (sda_in).

An integrated 153-bit packet required 42.96 μs to be received at an operating frequency (SCL) of 3.6 MHz. The throughput was approximately 3.5614 Mbps, which was approximately 8.15 times higher than that of the UART packet. When the edge computing unit received the integrated packet, the online linear regression, NN, and data monitoring units were triggered. The real and predicted humidities and temperatures are displayed in Fig. 17; the formulas for the relative humidity (RH) and temperature (T) conversions are provided in [14].

$$RH = 100 \times \frac{S_{RH}}{2^{16} - 1} \tag{5}$$

$$T\,(^oC) = -45 + 175 \times \frac{S_r}{2^{16} - 1} \tag{6}$$

where S_{RH} and S_r are the raw sensor outputs for humidity and temperature, respectively. The aforementioned formulas hold only when S_{RH} and S_r are represented in decimal form. For example, the hexadecimal code "6C25h" indicates an indoor temperature of 28.93 °C with the temperature conversion formula, as shown in (6).

Note that the Signal Tap logic analyzer, which handles conversion, was used to capture the hexadecimal code. The real sensing value could be obtained directly from the computer. Fig. 18 presents the real and predicted temperatures for comparison. The root-mean-square error is defined in (7), in which N is the number of data points, y_i is the i^{th} actual value, and $\hat{y}_i$is the i^{th} predicted value [23].

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} (y_i - \hat{y}_i)^2}{N}} \tag{7}$$

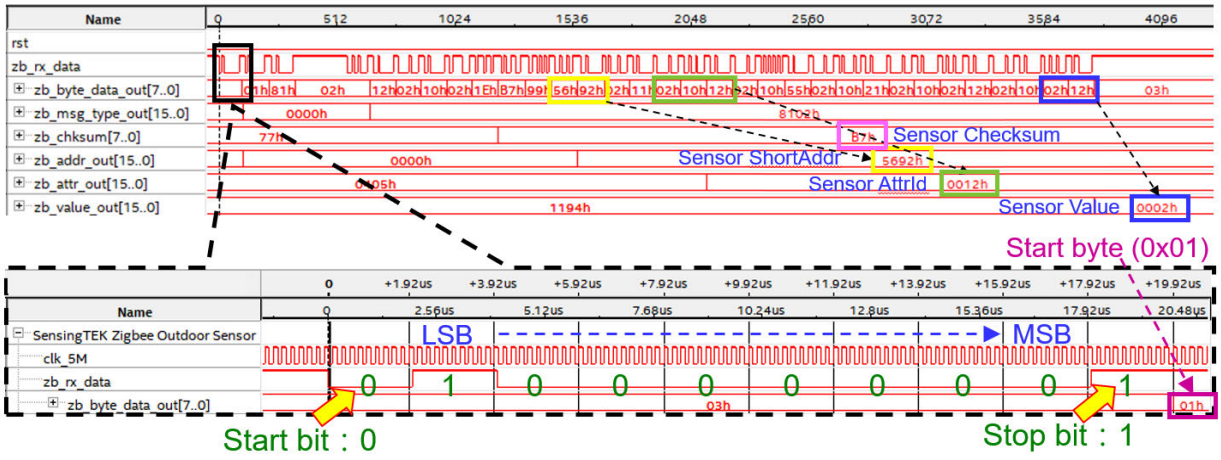

FIGURE 15. Analysis and verification of the measured UART packet.

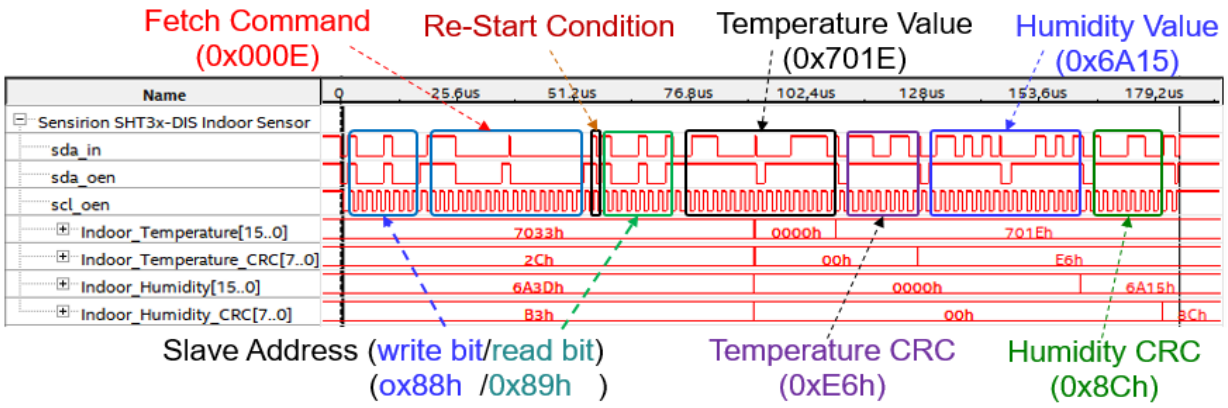

FIGURE 16. Analysis and verification of the measured I^2C packet of SHT31-DIS.

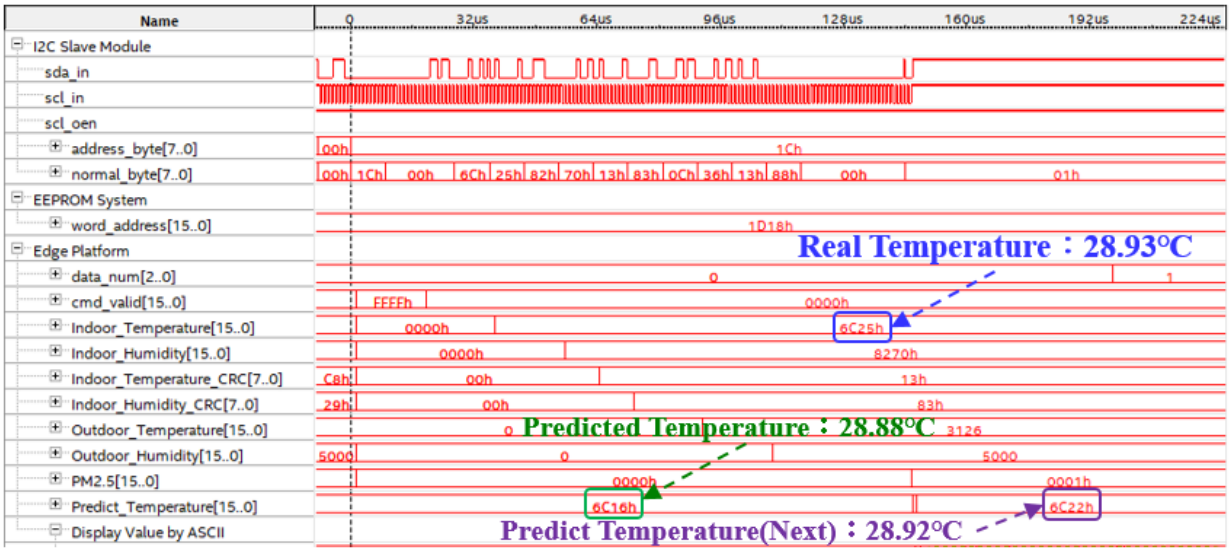

FIGURE 17. Real and predicted humidity and temperatures obtained using the RH and temperature conversion formulas.

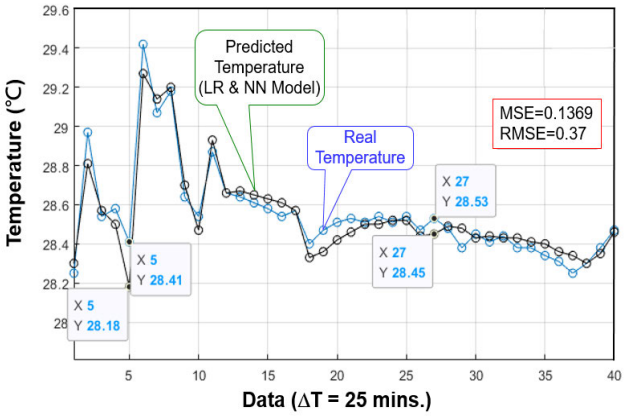

FIGURE 18. Real and predicted temperatures.

TABLE 3. Specifications of the ASIC of the front-end gateway system.

Performances	Specification
Technology	TSMC 90-nm 1P9M
Supplied Voltage (V)	3.3 / 1.0
Clock Frequency (MHz)	10 / 5 / 1
Power Consumption (mW)	0.6545
Gate Count (gates)	6,057
Chip area (mm^2)	0.8299×0.8297
Package	S/B-48

B. ASIC IMPLEMENTATION AND FUNCTIONAL VERIFICATION

We evaluated the performance of the front-end gateway ASIC in protocol conversion and data transmission. A sensor (SensingTek) collected data on temperature, humidity, and particulate matter 2.5 level [14]. The communication protocol of the wireless sensor was ZigBee,

and the onboard sensing device was a Sensirion SHT31-DIS device, which measured temperature and humidity and transmitted the measured data through the I^2C protocol. In the ASIC, debugging and verification were conducted using the NC-Verilog simulator and Verdi/nWave waveform viewer.

After the functional verification of the ASIC and the packet analysis of the I^2C master and UART, we synthesized the ASIC using a TSMC 90-nm CMOS cell-based process, and the clock frequency was set as 10 MHz. The Synopsys IC Compiler was used to plan the chip layout of the front-end gateway system. After the ASIC passed the design rule check (DRC) and layout versus schematic (LVS) verifications, the chip of the front-end gateway system was imaged (Fig. 19). Table 3 presents the specifications of the ASIC of the proposed front-end gateway system. Fig. 20 shows the post-layout simulated waveforms of the front-end gateway system, which were captured using the Verdi/nWave waveform viewer. The chip layout was correctly fabricated, as indicated by the waveforms.

A PC-based logic analyzer from the Acute TravelLogic series (TravelLogic TL2236), which was newly released on

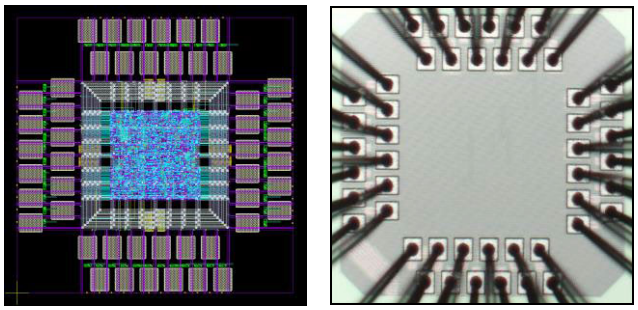

FIGURE 19. ASIC of the front-end gateway system: (a) chip layout and (b) chip photograph.

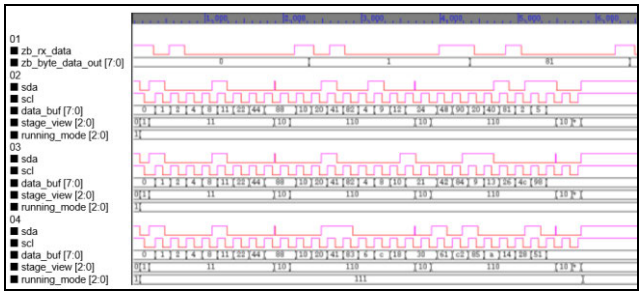

FIGURE 20. Post-layout simulated waveforms of the front-end gateway system.

TABLE 4. Throughput results for the wireless sensor, onboard sensor, and I^2C integrated packet.

Parameter	Wireless Sensor	Onboard Sensor	I^2C Integrated Packet
Protocol	UART	I^2C	I^2C
Operating frequency (MHz)	5	0.5	3.6
Transmitted Bits (bits)	320	90	153
Throughput (Mbps)	0.4478	0.4877	3.5614

the market, was used to measure the output waveform of the I^2C packet. The waveform results indicated that the I^2C FSM was working correctly. Fig. 21 displays the captured waveform of the I^2C packet at a clock frequency of 10 MHz and an SCL frequency of 0.5 MHz. Subsequently, two input signals of the integrated packet, namely SDA and SCL, were captured by the Signal Tap logic analyzer in a post-layout simulation and measured using the logic analyzer (TravelLogic TL2236). Fig. 22 depicts the measured SDA and SCL signals of the integrated packet. The proposed ASIC was working correctly. A summary of the throughput results of the wireless sensor, onboard sensor, and I^2C integrated packet are presented in Table 4. The highest throughput was approximately 3.5614 Mbps in the high-speed mode under an operating frequency (SCL) of 3.6 MHz.

The destination of this study is not only to integrated the proposed front-end gateway system with serial

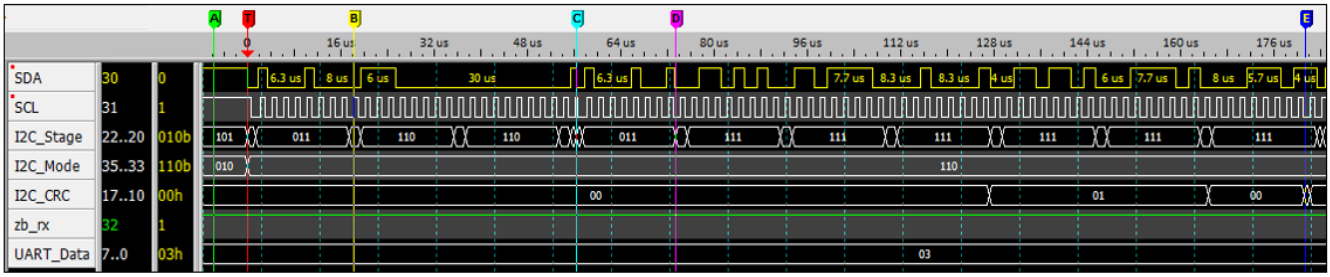

FIGURE 21. Captured waveforms of the I²C packet at clock frequency of 10 MHz and an SCL frequency of 0.5 MHz.

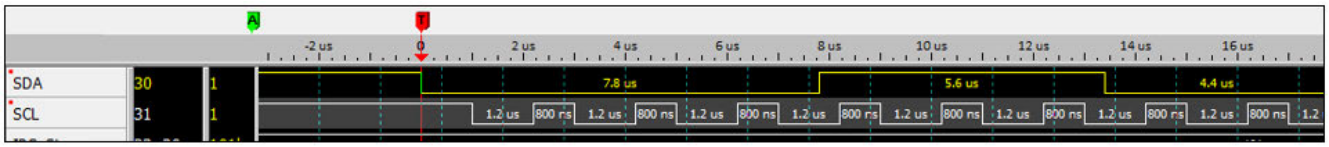

FIGURE 22. Measured SDA and SCL signals of the intergrated packet that were obtained from the logic analyzer (TravelLogic TL2236.

TABLE 5. Performance results of the proposed front-end gateway asic with serial communication protocol conversion and previously proposed methods.

Parameter	This work	[7] (2018)	[18] (2019)	[24] (2021)	[25] (2020)
Protocol Conversion	UART → I²C	SPI → I²C	I²C	UART → I²C	I²C
Implementation Device	FPGA DE-10	Xilinx ISE 14.7	XC6SLX9	ESP 32	AT89C52RC
Clock Frequency	10 MHz	50 MHz	50 MHz	–	24 MHz
SCL Frequency	3.6 MHz	~ MHz	1.0 MHz	115.2 kHz	100 kHz
Data Collision	Arbitration CSMA/CA	–	–	–	–
Memory	EEPROM	FIFO (queue)	Register (256)	–	EEPROM
ASIC	Yes	×	×	×	×

communication protocol conversion, but also to implement it into an ASIC, which is used to accelerate the conversion speed and to enhance the system's robustness. This is the only ASIC compared against relevant methods proposed in previous studies (Table 5). The methods of [7] and [24] use protocol conversion, and the methods of [18] and [25] are designed for data communication through the I^2C protocol. These methods were implemented on commercial devices that were different from our device, which was designed with DE10-Standard development kit. Notably, only this study can prevent data collision through an arbitration process based on the CSMA/CA protocol. Our device has a higher maximum SCL (approximately 3.6 MHz) compared against those methods proposed in the aforementioned studies. The EEPROM communicates with the proposed ASIC through the I^2C protocol and stores sensing data to prevent data loss caused by power outages. The throughputs of the wireless sensor, onboard sensor, and I^2C integrated packet were high at 447 kbps, 487.7 kbps, and 3.5614 Mbps, respectively (Table 4). Thus, the designed ASIC can be used in various applications.

V. CONCLUSION

In this paper, a front-end gateway system with serial communication protocol conversion and an edge computing platform is proposed. This system is useful for deep learning applications on IoT networks that not only receive and store sensing data by using an FPGA board but also leverage edge computing for prediction. The front-end gateway system is composed of a data control unit and a serial communication PCU with the I²C and UART protocols. The proposed gateway system also accelerates integrated packet transmission through arbitration based on the CSMA/CA protocol to prevent collision. The highest SCL frequency of the I²C slave is approximately 3.6 MHz for sending integrated packets to the edge computing platform. Furthermore, the SCL frequency of the I²C master can be reduced to 0.2 MHz in edge computing platforms for real-time data monitoring and data storage in EEPROM. The simulation results indicate that the NN of the proposed system outperforms simple linear regression. In this study, a front-end gateway ASIC was fabricated using the TSMC 90-nm 1P9M CMOS process. The power consumption, gate count, and chip area of this circuit were 0.6545 mW,

6,057 gates, and 0.8299×0.8297 mm^2, respectively, at 3.3 V and a clock frequency of 10 MHz. Our designed ASIC can be added to the front-end gateway system to enable this system to achieve faster communication, a smaller size, and greater robustness.

ACKNOWLEDGMENT

The authors are grateful to the Taiwan Semiconductor Research Institute (TSRI), Taiwan, for fabricating the test chip. This manuscript was edited by Wallace Academic Editing.

REFERENCES

[1] M. E. M. Campista, L. H. M. K. Costa, and O. C. M. B. Duarte, "Improving the multiple access method of CSMA/CA home networks," in *Proc. IEEE Consum. Commun. Netw. Conf. (CCNC)*, Las Vegas, NV, USA, Jan. 2006, pp. 645–649.

[2] J. Peng and L. Cheng, "Revisiting carrier sense multiple access with collision avoidance (CSMA/CA)," in *Proc. 40th Annu. Conf. Inf. Sci. Syst.*, Princeton, NJ, USA, Mar. 2006, pp. 1236–1241.

[3] M. Vallérian, F. Hutu, G. Villemaud, B. Miscopein, and T. Risset, "A parallel unbalanced digitization architecture to reduce the dynamic range of multiple signals," *Radio Sci.*, vol. 51, no. 5, pp. 411–420, May 2016.

[4] H.-H. Choi, J.-M. Moon, I.-H. Lee, and H. Lee, "Carrier sense multiple access with collision resolution," *IEEE Commun. Lett.*, vol. 17, no. 6, pp. 1284–1287, Jun. 2013.

[5] H. Watanabe, T. Kondo, and T. Ohigashi, "Implementation of platform controller and process modules of the edge computing for IoT platform," in *Proc. IEEE Int. Conf. Pervasive Comput. Commun. Workshops (PerCom Workshops)*, Kyoto, Japan, Mar. 2019, pp. 407–410.

[6] T. Kondo, H. Watanabe, and T. Ohigashi, "Development of the edge computing platform based on functional modulation architecture," in *Proc. IEEE 41st Annu. Comput. Softw. Appl. Conf. (COMPSAC)*, Turin, Italy, vol. 2, Jul. 2017, pp. 284–285.

[7] D. Trivedi, A. Khade, K. Jain, and R. Jadhav, "SPI to I2C protocol conversion using verilog," in *Proc. 4th Int. Conf. Comput. Commun. Control Autom. (ICCUBEA)*, Pune, India, Aug. 2018, pp. 1–4.

[8] *I2C-Bus Specification and User Manual (UM10204)*, NXP Semiconductors Taiwan Ltd., Kaohsiung, Taiwan, 2021, pp. 9–20.

[9] R. S. S. Kumari and C. Gayathri, "Interfacing of MEMS motion sensor with FPGA using I2C protocol," in *Proc. Int. Conf. Innov. Inf., Embedded Commun. Syst. (ICIIECS)*, Coimbatore, India, Mar. 2017, pp. 1–5.

[10] *DE10-Standard Board: Cyclone V*, Terasic headquarter, Hsinchu, Taiwan, 2018. [Online]. Available: https://www.terasic.com.tw/cgi-bin/page/archive.pl?Language=English&CategoryNo=167&No=1081

[11] C. Wang, T. Jiang, and Q. Zhang, *ZigBee Network Protocols and Applications*, 1st ed. New York, NY, USA: Auerbach, Mar. 2014.

[12] *ZigBee IoT Gateway Control Bridge (JN-AN-1223)*, NXP Semiconductors Taiwan Ltd., Kaohsiung, Taiwan, Aug. 2017, pp. 27–39.

[13] D. Jahier Pagliari, E. Macii, and M. Poncino, "Zero-transition serial encoding for image sensors," *IEEE Sensors J.*, vol. 17, no. 8, pp. 2563–2571, Apr. 2017.

[14] *Datasheet SHT3x-DIS: Humidity and Temperature Sensor*, Sensirion Company, Stafa, Switzerland, Feb. 2019, pp. 7–15.

[15] W. W. Peterson and D. T. Brown, "Cyclic codes for error detection," *Proc. IRE*, vol. 49, no. 1, pp. 228–235, Jan. 1961.

[16] B. Eswari, N. Ponmagal, K. Preethi, and S. G. Sreejeesh, "Implementation of I2C master bus controller on FPGA," in *Proc. Int. Conf. Commun. Signal Process.*, Melmaruvathur, India, Apr. 2013, pp. 678–681.

[17] M. Sukhanya and K. Gavaskar, "Functional verification environment for I2C master controller using system verilog," in *Proc. 4th Int. Conf. Signal Process., Commun. Netw. (ICSCN)*, Chennai, India, Mar. 2017, pp. 1–6.

[18] C. Liu, Q. Meng, T. Liao, X. Bao, and C. Xu, "A flexible hardware architecture for slave device of I2C bus," in *Proc. Int. Conf. Electron. Eng. Informat. (EEI)*, Nanjing, China, Nov. 2019, pp. 309–313.

[19] D. Wang, Y. Gao, and Z. Tian, "One-variable linear regression mathematical model of color reading and material concentration identification," in *Proc. Int. Conf. Smart City Syst. Eng. (ICSCSE)*, Changsha, China, Nov. 2017, pp. 119–122.

[20] Y. Sai, R. Jinxia, and L. Zhongxia, "Learning of neural networks based on weighted mean squares error function," in *Proc. 2nd Int. Symp. Comput. Intell. Design*, Changsha, China, vol. 1, Dec. 2009, pp. 241–244.

[21] A. Chawla, P. Babu, T. Gawande, E. Aumayr, P. Jacob, and S. Fallon, "Intelligent monitoring of IoT devices using neural networks," in *Proc. 24th Conf. Innov. Clouds, Internet Netw. Workshops (ICIN)*, Paris, France, Mar. 2021, pp. 137–139.

[22] *Intel Quartus Prime Standard Edition User Guide: Debug Tools, v. 2018.09.24*, Intel Corp., Santa Clara, CA, USA, 2018.

[23] W. Zhu, X. Liu, and Z. Zhu, "Optimized air quality prediction model based on neural network," in *Proc. Int. Conf. Comput. Eng. Appl. (ICCEA)*, Guangzhou, China, Mar. 2020, pp. 565–568.

[24] Y. S. Elshakhs and M. Pudzs, "Communication system for standardized multipurpose CNC machine," in *Proc. IEEE 9th Workshop Adv. Inf., Electron. Electr. Eng. (AIEEE)*, Riga, Latvia, Nov. 2021, pp. 1–5.

[25] Y. Chen, F. Xiao, L. Yu, and P. Cui, "Design of multi-line elastic belt conveying control system for knitting machine based on I2C protocol," *IEEE Access*, vol. 8, pp. 51803–51809, 2020.

GUO-MING SUNG (Member, IEEE) was born in Zhanghua, Taiwan, in 1963. He received the B.S. and M.S. degrees in biomedical engineering from Chung Yuan Christian University, Taoyuan, in 1987 and 1989, respectively, and the Ph.D. degree in electrical engineering from National Taiwan University, Taipei, Taiwan, in 2001.

In 1992, he joined the Division of Engineering and Applied Sciences, National Science Council, Taiwan, where he became an Associate Researcher, in 1996. Since 2001, he has been with the Department of Electrical Engineering, National Taipei University of Technology, where he is currently a Professor and the Chairperson. His research interests include magnetic/pressure sensors, motor control IC, ethernet/USB/RS485 bridge IC, AI chip, ADC/DAC ICs, radio-frequency harvester IC, mixed-mode IC, and the Internet of Things (IoT) and its applications.

LI-FEN TUNG received the M.S. degree in electrical engineering from the National Taipei University of Technology, in 2019, where she is currently pursuing the Ph.D. degree with the Department of Electrical Engineering. Since 2000, she has been with semiconductor industry. She is involved in semiconductor devices, motor control, and the IoT with emphasis on practical applications.

CHIH-JUNG HUANG was born in Nantou, Taiwan, in 1993. He received the B.S. degree in electrical engineering from National Chi Nan University, in 2016, and the M.S. degree in electrical engineering from the National Taipei University of Technology, Taipei, Taiwan, in 2022. His research interests include intelligent internet of sensors, communication protocols, and artificial intelligence.

CHIH-PING YU was born in Keelung, Taiwan. He received the B.S. degree in electrical engineering from the National Taiwan University of Science and Technology, Taiwan, in 1989, and the M.S. degree in electrical engineering from Washington University in St. Louis, USA, in 1995. He is currently an Associate Professor with the Department of Electrical Engineering, National Taipei University of Technology. His research interests include magnetic sensor and its applications, analogue filter circuits, mixed-mode circuit design, digital communication, and the IoT applications.