IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 13 June 2024, accepted 2 July 2024, date of publication 22 July 2024, date of current version 12 August 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3432175

RESEARCH ARTICLE

Ethernet Packet-to-RS485 Data Transfer Bridge Application-Specific Integrated Circuit Incorporating Transmission Control Protocol and Static Random-Access Memory

GUO-MING SUNG, (Member, IEEE), CHUN-TING LEE, ZHI-LUN YOU, 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 (NSTC), Ministry of Science and Technology, Taiwan, under Contract MOST 111-2221-E-027-122.

ABSTRACT This paper proposes a high-throughput application-specific integrated circuit (ASIC) designed for bridging Ethernet packets and RS485 data, featuring a transmission control protocol (TCP) and static random-access memory (SRAM). The bridge ASIC complies with a 1-gigabit media-independent interface (GMII) and media-independent interface protocol standards. As the Ethernet receiver (RX) receives an Ethernet packet from GMII, it stores the TCP header and payload data in the SRAM. Next, the RS485 module reads the data stored in the SRAM for transmission to a computer through an RS485-to-universal serial bus cable. The Ethernet transmitter (TX) reads the TCP header and payload data from the SRAM and forms a complete Ethernet packet by adding the source address, destination address, and Internet protocol header. Passing through the GMII device, the complete Ethernet packet is sent to the Ethernet port, which is connected to a computer with an RJ45 network cable. This study was validated using NC-Verilog software and a field-programmable gate array board (DE10-Standard). After successful verification, the ASIC was fabricated using the Taiwan Semiconductor Manufacturing Company 0.18-μm complementary metal–oxide semiconductor process. The simulated and measured results indicate that the throughput, processing latency, power consumption, gate count, and chip area are 844.88 Mbps, 139.31 μs, 134.79 mW, 69287, and 1.19×1.19 mm^2, respectively, under an operating frequency of 125 MHz, on-chip SRAM of 256 bytes, and power supply voltage of 1.8 V. The key contribution not only reduces the size of the RS485-to-Ethernet module but also improves the performance and saves the chip cost.

INDEX TERMS Ethernet packet, RS485 data, transmission control protocol, static random-access memory, media-independent interface (MII), FPGA board, ASIC.

I. INTRODUCTION

The rapid development of Industry 4.0, the Internet of Things, and car networking have elevated the role of Ethernet as a reliable protocol suite. In industrial networks, specialized communication interfaces such as RS-232 and RS-485 serve as the media for data transmission in the absence of standardized communication protocols. Ethernet is a popular communication protocol that integrates these sensing devices in industrial networks. The primary objectives of industrial Ethernet include not only enhancing the stability of a network but also ensuring compatibility with existing communication protocols such as Modbus TCP/IP, Ethernet/IP, EtherCAT, PROFINET, POWERLINK, and SERCOS III [1], [2], [3].

The associate editor coordinating the review of this manuscript and approving it for publication was Ilaria De Munari.

More stringent and detailed planning is necessary for network reliability and information security during data transmission and production line control.

In industrial applications, Ethernet is commonly deployed as the communication system to integrate various sensing equipment. However, most industrial communication systems and linear accelerators rely on serial communication interfaces such as RS-232, RS-422, or RS-485 for control and monitoring [4]. Among these, RS-485 is most frequently used in remote control communication systems. These systems often encounter difficulties in transmitting sensing signals over transmission lines due to the multiplicity of communication protocols tailored for specific sensing devices [5], [6]. A technology of RS485 over Ethernet developed in [7] enables the transmission of multiple serial TCP/IP data packets over Ethernet, thereby establishing wired connections between computers on a network. Consequently, packet conversion and transmission between Ethernet and RS-485 have gained prominence. They both contribute to an increase in the communication rate and an improvement in the reliability of data communications. The Ethernet-to-RS485 converter application-specific integrated circuit (ASIC) proposed in this paper not only improves communication features but also maintains compatibility with multiple protocols without necessitating changes to the original RS485 topology.

Conventional software-based protocol stacks require substantial CPU processing time when operating at full transmission rates, resulting in performance bottlenecks such as high latency and low throughput. Hardware design can be used to increase communication rates and shorten application-to-application latency. Hardware-based TCP offload engines (TOEs) have been proposed in [8] and [9] to improve throughput while maintaining support for TCP features. The flexibility and extendibility of field-programmable gate array (FPGA) implementations further augment these capabilities. Using TOE and kernel bypass techniques reduces latency to one-tenth of the latency between a Linux host without TOE and a Windows host. In addition, Ethernet-based embedded systems are required for a variety of applications that require high-speed data transmission over long distances. FPGA is a potential solution for the development of customized Ethernet-based embedded systems [10]. An FPGA-based W5500 Ethernet controller interface was developed as an alternative to TCP/UDP software stacks. This design constitutes a cost-effective FPGA-based TCP/UDP module, offering an alternative to commercially available TCP/UDP IP cores. Further development can be achieved by incorporating TCP mode support into the W5500 Ethernet controller interface. One study [11] proposed an Ethernet architecture for packet transformation and transmission between Modbus transmission control protocol (Modbus/TCP) and universal serial bus (USB) 3.0 using an FPGA development board. The bridge mentioned above achieves high throughput and low latency using the FPGA development board, which provides an ASIC solution with low power consumption. The robustness and convenience of the proposed ASIC are particularly advantageous [12]. This proposed architecture holds potential applications in plant automation.

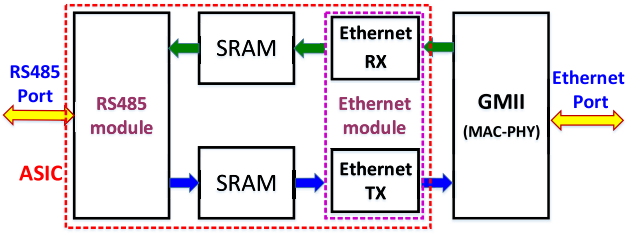

FIGURE 1. Proposed communication architecture of the bridge ASIC between ethernet and RS485 modules with SRAM cells.

Fig. 1 illustrates the proposed architecture for communication between Ethernet and RS485 modules. This architecture includes the gigabit media-independent interface (GMII); which is an interface between the medium access control (MAC) device and the physical layer (PHY); Ethernet RX, Ethernet transmitter (TX); SRAM; and RS485 module. When data must be transmitted from the RS485 port to the Ethernet port, the RS485 module receives the transmitted data, writes them to the SRAM, and calculates the data length. After the transmitted data have been fully stored in SRAM, the calculated data length is sent to the Ethernet TX, which waits for a transmission signal. Upon receiving this signal, the Ethernet TX sends the preamble in sequence, followed by the source and destination MAC addresses. A cyclic redundancy check (CRC) value is calculated simultaneously. Next, the IP checksum is calculated and verified. If the checksum is verified, the IP header is sent. Afterward, the stored data is sequentially read from the SRAM, the CRC value is checked, and a complete Ethernet packet is successfully transmitted from SRAM to the Ethernet port through the GMII module. Two SRAMs are used to address the frequency incompatibility between Ethernet (125 MHz) [13] and RS485. The classic cable communication standard RS485 allows the connection of simple devices to the TCP/IP network through an intermediary node. It connects multiple nodes over a twisted-pair bus, using differential signaling to reach distances of up to 1 km, with typical transmission rates between 9.6 and 115.2 kbps [14]. The remainder of this paper is organized as follows: Section II describes the system architecture of the proposed Ethernet and RS485 modules. Section III presents the simulated results and functional validation. Section IV provides the ASIC implementation and measurement results of the proposed bridge system with TCP and SRAM. Finally, Section V presents conclusions.

II. SYSTEM ARCHITECTURE OF THE PROPOSED ETHERNET AND RS485 MODULES

The GMII provides multiple independent TX and RX communication channels between the proposed bridge ASIC and the physical layer (PHY) chip, with two reference clocks designated for transmitting and receiving. The GMII uses 8-bit metadata transmission with an operating clock of

125 MHz, enabling a transmission rate of up to 1000 Mbps but retains compatibility with the 10- and 100-Mbps transmission rates as specified by the media-independent interface [15]. Notably, the GMII is responsible for several connected signals, including GTXCLK (clock signal for gigabit transmission at 125 MHz), TXCLK (clock signal for 10/100 Mbps), TXER (transmitter error), TXEN (TX enabled), TXD[7..0] (data transmitted), RXCLK (clock signal of receiver), RXER (receiver error), TXEN (RX enabled), and RXD[7..0] (data).

A. ETHERNET TX MODULE

The Ethernet TX module reads the data stored in the SRAM received from the RS485 RX and packages it into an Ethernet packet for transmission to the GMII module at an operating frequency of 125 MHz. Fig. 2 depicts a block diagram of the proposed Ethernet TX mode, which is composed of an Ethernet TCP/IP module, an Ethernet MAC module, and a CRC module.

Fig. 3 provides a flowchart detailing the operational sequence of the Ethernet TX module. When the Ethernet TCP/IP module is started, both the data received from the RS485 RX and the corresponding data length are stored in the SRAM. The calculated data length and the IP checksum in the IP header are computed based on these received data.

After the Ethernet TX module receives both the data stored in SRAM and the trigger signal, the Ethernet MAC module starts its transmission function. First, a 7-byte preamble, a 1-byte start frame delimiter (SFD), a TXEN signal set to high (1), and a TXER signal set to low (0), are sent. Next, the destination address (6 bytes) and the source address (6 bytes) are sent simultaneously to calculate the frame check sequence (FCS) and to activate the Ethernet TCP/IP module using the trigger signal. In the Ethernet TCP/IP module, the IP header and IP checksum are calculated and sent to the subsequent Ethernet MAC module. The TCP header and data stored in the SRAM are also sent to the Ethernet MAC module. Upon complete reception of those values from the Ethernet TCP/IP module, the Ethernet MAC module calculates a 4-byte FCS and appends it to the Ethernet packet. If the calculated FCS is incorrect, the TCP header is retransmitted by Ethernet TCP module. If the calculated FCS is corrected, then the Ethernet packet is sent with control flags, SYN and ACK. Next, a retransmission is initiated with an uncorrected ACK signal. If the received ACK signal is corrected, then the TXEN signal is set to low (0), finalizing the Ethernet packet. Finally, a complete Ethernet packet has been successfully sent from the Ethernet TCP/IP module to the Ethernet MAC module. After the transmission of the packet is finished, the packet must be stored until it is acknowledged by the recipient [15].

B. ETHERNET RX MODULE

The Ethernet RX module receives and decomposes the Ethernet packets transmitted through the GMII module. Next, the decomposed TCP header and transmitted data are stored in

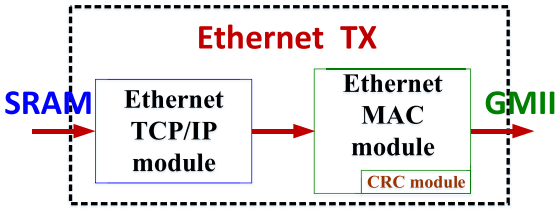

FIGURE 2. Proposed ethernet TX module.

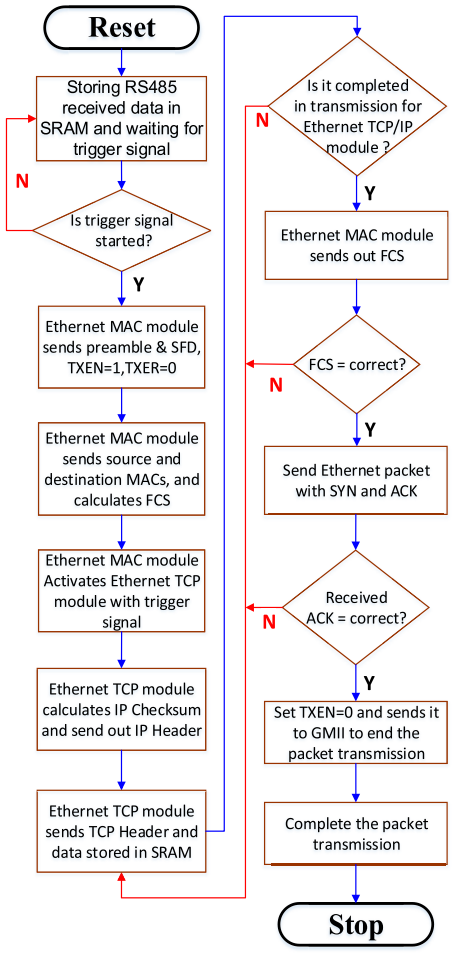

FIGURE 3. Operational flowchart of ethernet TX module.

the subsequent SRAM at an operating frequency of 125 MHz. Because the RX_DV and RX_ER signals, which are transmitted from the GMII module, are set to high (1) and low (0), respectively, the Ethernet RX module receives the RXD[7..0] data and ends the receiving process when the RX_DV signal becomes low (0). If the RX_ER signal changes to high (1), the Ethernet RX module receiving process is interrupted, and the incomplete packet is discarded.

The Ethernet RX module starts to receive the data of RXD[7..0] by capturing the preamble. If the first 8 bits of data of the preamble match the SFD of the incoming Ethernet packet, the Ethernet TX module receives the destination MAC address. If the received destination MAC address matches the

preset destination, it continues to receive the source MAC address. After the received source MAC address is verified, the Ethernet RX module receives the IP header data and calculates the IP checksum. If the IP checksum matches the calculated value, the RX module is ready to receive the TCP header and data. Simultaneously, the SRAM write signal is started, and the data are stored in the SRAM. After the stored data are completely received, the RX module captures the FCS and deactivates the SRAM write signal. When the RX_DV signal is set to low (0), the receiving process is stopped, and the next Ethernet packet is ready to be received. In other words, a retransmission is initiated when those control parameters, namely RX_DV, RX_ER, Count, and IP Checksum, do not meet the specified criteria. Fig. 4 provides a flowchart detailing the operational sequence of the Ethernet RX module.

C. SYSTEM ARCHITECTURE OF RS485 MODULE

Fig. 5 illustrates a block diagram of the proposed RS485 module, which is composed of an RS485 TX module and an RS485 RX module. The baud rate of the RS485 module is 115,200 bps, enabling the transmission or reception of 1 bit of data every 8600 ns. To receive or transmit all 11 bits of data, appropriately 9640 ns (11 × 8600 ns) are required. The operating frequency of the Ethernet module is 125 MHz. Using a counter to count the required cycles for 1-bit data transmission or reception, approximately 1075 cycles (8600 ns × 125 MHz) are required. Thus, a 16-bit counter is used to complete the counting for 11 bits.

The function of the RS485 TX module is to convert parallel data, which are stored in SRAM, into serial data by using the universal asynchronous receiver/transmitter (UART) protocol. Fig. 6 is a flowchart outlining the operational sequence of the RS485 TX module. Initially, the RS485 TX module receives the RS485 TX enable signal, denoted as RS485_EN, from the SRAM. If RS485_EN signal is high (1), the RS485 RX module begins its operation. First, the counter is reset, and the initial parallel data are read from the SRAM. Next, the read parallel data stored in the TX module are converted into serial data according to the counter's value. That is, the RS485 TX module sends 1 bit of data every 1075 cycles, requiring a total of 11,825 cycles for 11 bits. When the counter's value reaches 11,825, the conversion function is complete. The final step involves verifying that the length of the transmitted serial data aligns with the length of the received parallel data from the SRAM. If the lengths do not match, the counter is reset, and the subsequent SRAM data are retransmitted for conversion. This conversion process continues until the lengths align, signifying the successful transmission of data by the RS485 TX module.

The RS485 RX module receives the serial data encoded with the UART protocol and converts them into 8 bits of parallel data, which are subsequently stored in the SRAM. Fig. 7 is an operational flowchart of the RS485 RX module. When the RS485 RX module receives serial data encoded with UART from a personal computer (PC), it sequentially

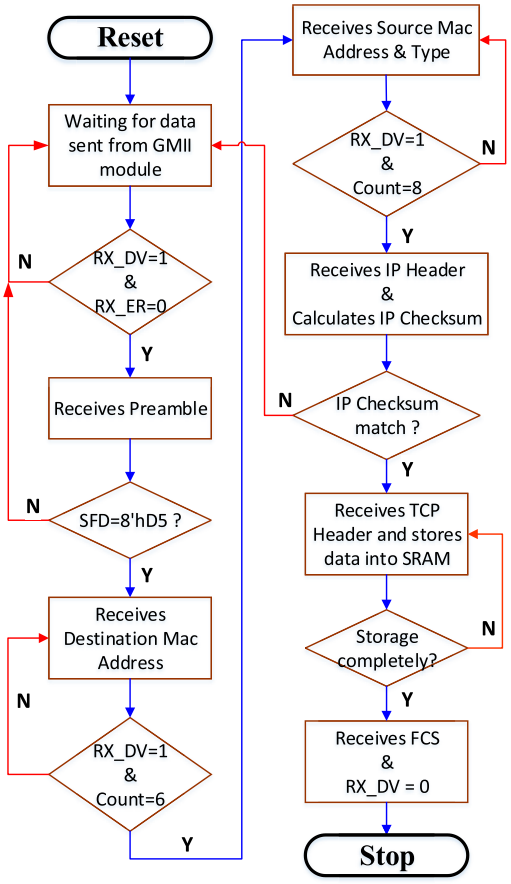

FIGURE 4. Operational flowchart of ethernet RX module.

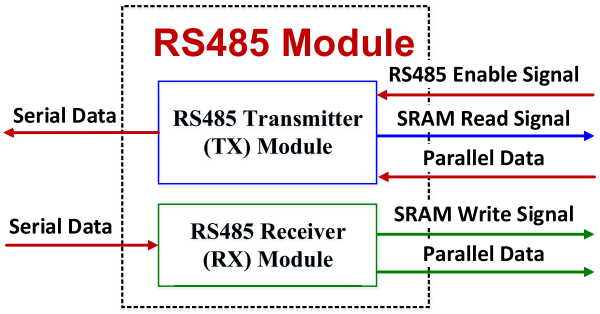

FIGURE 5. Block diagram of proposed RS485 module, which is composed of RS485 TX and RX modules.

assembles the bits into an 11-bit frame, a process akin to that of the RS485 RX module. As the counter reaches 2150, 3225, 4300, 5375, 6450, 7525, 8600, and 9675, the RS485 RX module receives the first bit (0), second bit (1), third bit (2), fourth bit (3), fifth bit (4), sixth bit (5), seventh bit (6), and eighth bit (7), respectively. Next, the RS485 RX module writes the assembled 8 bits of data to the SRAM when the counter value lies between 9676 and 11,825. Simultaneously, the module calculates the length of the written data to confirm all 8 bits have been successfully written to the SRAM. If the

transmission is not completed, the counting procedure is restarted by resetting the counter's value.

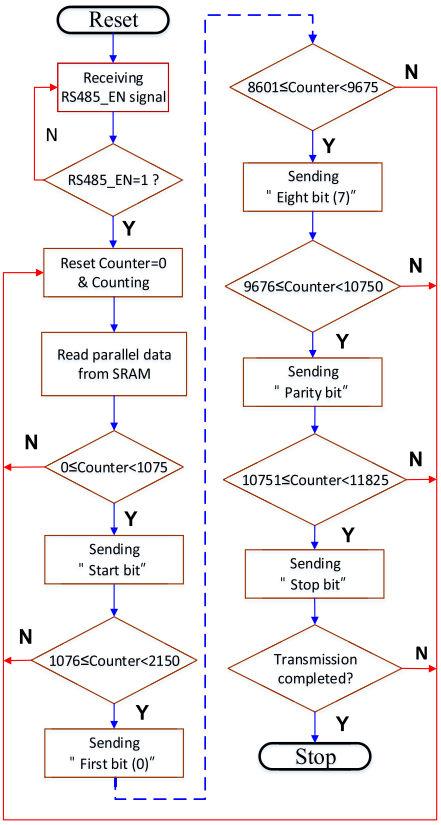

FIGURE 6. Flowchart of the operation of RS485 TX module.

The SRAM considered for the proposed design is an on-chip memory fabricated using the TSMC 0.18-μm complementary metal–oxide–semiconductor (CMOS) process, offers storage of approximately 256 × 8 bits, and operates at a frequency of 125 MHz. The Ethernet RX and RS485 RX modules write received data to the SRAM, whereas the Ethernet TX and RS485 TX modules read the data stored in the SRAM. Several input and output signals are integral to this operation, including: 8-bit data received (data[7..0]), write address (Wraddress[7..0]), write enabled (Wr_en), read address (Rdaddress[7..0]), read enabled (Rd_en), system clock (CLK), and 8-bit output data (Q[7..0]).

D. THROUGHPUT ANALYSIS

The proposed bridge ASIC can handle a maximum packet length that comprises an 8-byte preamble, a 14-byte MAC address, a 20-byte IP header, a 20-byte TCP header, a 236-byte maximum data, and a 4-byte FCS. The maximum length of a packet is 302 bytes. For a single data packet, which comprises a 20-byte TCP header and a maximum of 236 bytes of data, the total length amounts to 256 bytes. Fig. 8 presents the simulated propagation time required for transmitting the TCP header and data in Ethernet RX mode. The propagation time is approximately 2424 ns at

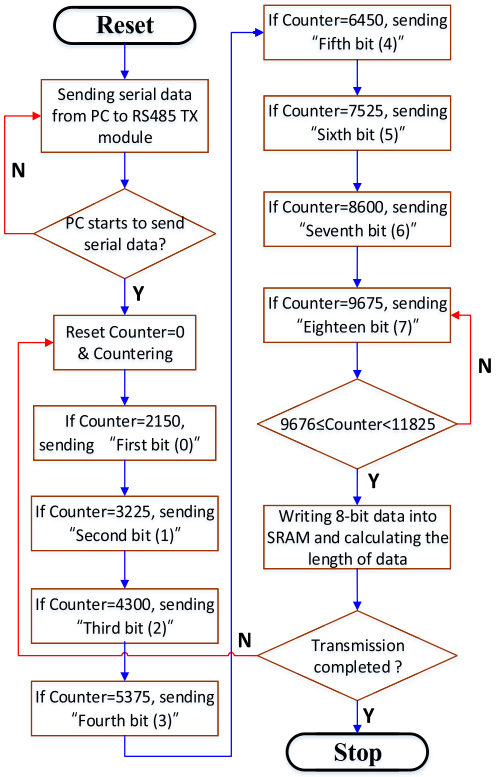

FIGURE 7. Operational flowchart of RS485 RX module.

an operating frequency of 125 MHz. The data throughput in this mode is calculated to be 844.88 Mbps (256 × 8 bits/2424 ns). In RS485 RX mode, the data throughput is roughly 0.0975 Mbps (2048 bits/21,006,898 ns) at an RS485 baud rate of 115,200 bps.

III. SIMULATED RESULTS AND FUNCTIONAL VALIDATION

The designed functions were simulated and validated using the NC-Verilog software. Two blocks are simulated to meet the design requirements: data transmission from the GMII module to the RS485 TX module and data transmission from the RS485 RX module to the GMII module.

FIGURE 8. Simulated propagation time for transmitting TCP header and data.

A. FROM GMII MODULE TO RS485 TX MODULE

The Ethernet RX module receives an Ethernet packet transmitted from the PHY. Fig. 9 illustrates the reception of an Ethernet packet transmitted from the GMII module. The clock signal (phy_clk) operates at 250 MHz, the driving

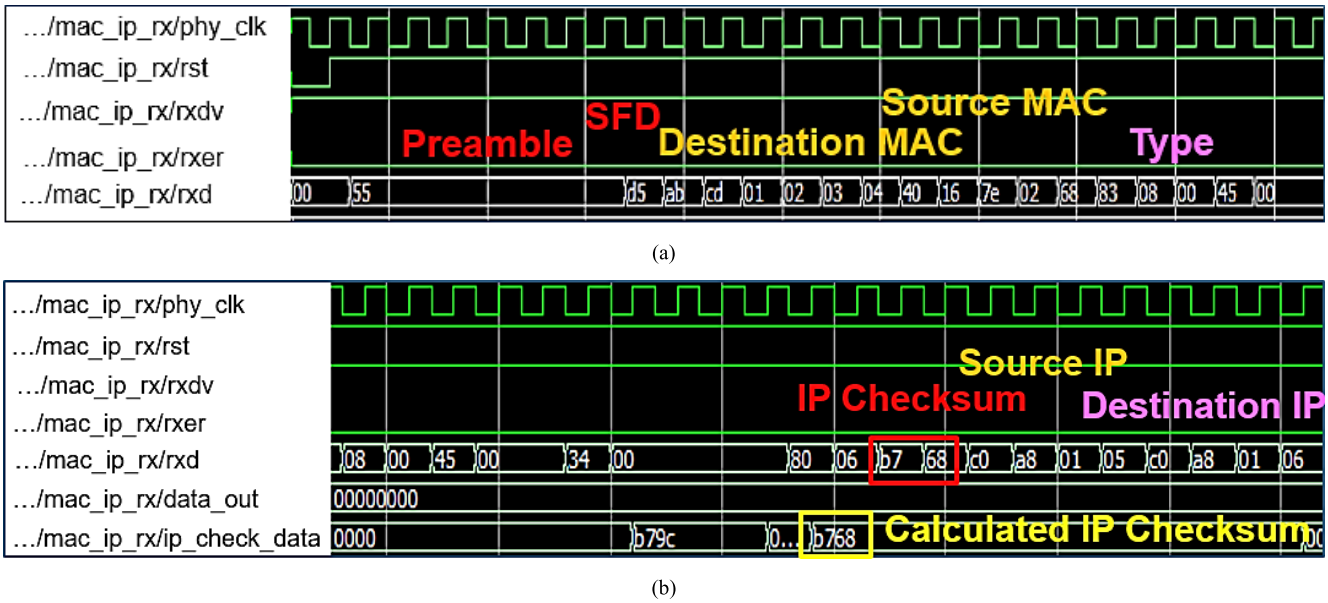

FIGURE 9. Ethernet RX module receives ethernet packet transmitted from GMII module with NC-Verilog software. (a) First section; (b) second section.

signal (rxdv) is set to high (1), the error signal (rxer) is set to low (0), and each received data unit (rxd) comprises 8 bits. As illustrated in Fig. 9(a), the Ethernet RX module first captures a 7-byte preamble and a 1-byte SFD, which are represented as the hexadecimal values "55" and "d5," respectively. Next, a 16-byte data string is received, consisting of a 7-byte destination MAC address, a 7-byte source MAC address, and a 2-byte type field. Specifically, the destination MAC address is "ab:cd:01:02:03:04," the source MAC address is "40:16:7e:02:68:83," and the type field reads "08:00." As illustrated in Fig. 9(b), upon receiving the IP header, the Ethernet RX module starts to calculate the IP checksum ("b7:68") in the data signal (rxd). The calculated IP checksum ("b768") appears in the check data signal (ip_check_data) and matches the value in the data signal. Next, the module captures a 4-byte source IP address ("c0:a8:01:05") and a 4-byte destination IP address ("c0:a8:01:06").

Fig. 10 illustrates the process wherein the Ethernet TX module receives the TCP header and stores the TCP data in the SRAM. When the Ethernet RX module receives the TCP header, the write signal (SRAM_W_EN) is activated and changes to high (1). The received data (tcp_data) are then stored in the SRAM at locations specified by the SRAM_ADDRESS. After TCP data transmission is complete, the write signal is terminated by setting SRAM_W_EN to low (0). Simultaneously, the system prepares to transmit the FCS and to activate the RS485 TX module by setting RS485_EN to high (1). Afterwards, the start signals of the RS485 TX module (RS485_start) transitions to high (1), triggering the counter.

In the RS485 TX module, data retrieval from the SRAM is initiated by setting the read signal of the SRAM (SRAM_read) to high (1). The selected SRAM data (DI) are serially transmitted to the RS485 TX module according to the selected SRAM address (SRAM_read_add). When the quantity of data read from SRAM matches that stored in SRAM, both the start signal of RS485 TX module (RS485_start) and the read signal of the SRAM (SRAM_read) transition to low (0). The counter is stopped and awaits the next Ethernet packet. Fig. 11 illustrates the read and transmission functions between the RS485 TX module and the SRAM. The read data comprise 11 bits: a 1 start bit (0), 8 bits of data (D0−D7), a 1 parity bit (1), and a 1 stop bit (1). For instance, if the read address of the SRAM is set to "1," the selected data "fa (1111,1010)" are serially transmitted to the RS485 TX module as "0, 01011111, 1, 1."

B. FROM RS485 RX MODULE TO GMII MODULE

When the RS485 RX module receives the RS485 serial data (RO) transmitted by the client, the data are stored in an 8-bit register according to the counter signal (counter). However, the start bit, parity bit, and stop bit are ignored during this process. When the writing signal (SRAM_WRITE) of SRAM is set to high (1), the register data are transferred to the SRAM (SRAM_DATA) according to the SRAM address (DATA_counter) and the length of stored data (TCP_len) is calculated. If the serial data are transmitted completely, the writing signal (SRAM_WRITE) of the SRAM transitions to low (0), thereby terminating the write function. Then the calculated length of stored data is sent to the Ethernet TX module. Fig. 12 illustrates the reception of the serial data transmitted from the RS485 RX module, which is then stored in the SRAM, with the NC-Verilog software.

Next, the Ethernet TX module is started by setting the trigger signal (inv_top_tcp_tx) to high (1). The Ethernet

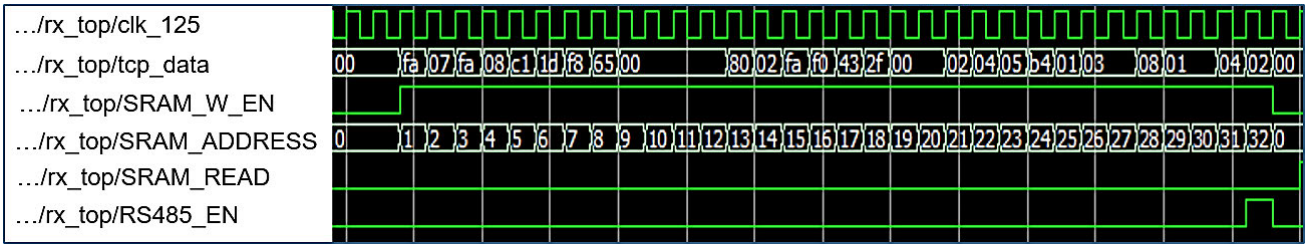

FIGURE 10. Ethernet TX module received TCP header and stores TCP data in SRAM.

FIGURE 11. Read and transmission functions between RS485 TX module and SRAM. Data comprise a 1-bit start bit, 8 bits of data (D0−D7), a 1-bit parity (1), and a 1-bit stop bit (1).

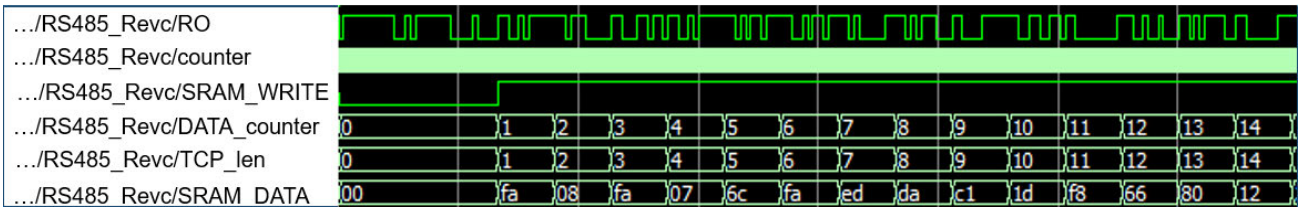

FIGURE 12. RS485 RX module transmits the RS485 serial data and stores them in SRAM.

MAC module starts to send a 7-byte preamble, a 1-byte SFD, a 6-byte destination MAC address, a 6-byte source MAC address, and a 2-byte type field. When the Ethernet MAC module sends the destination MAC address, the CRC module calculates the FCS, starting with an initial value of "32'hFFFFFFFF." Then a trigger signal (tcp_trig) is started by the Ethernet MAC module to engage the Ethernet TCP/IP module for the calculation of the IP header and IP checksum. If the IP checksum is correct, the Ethernet TCP/IP module sequentially sends the source IP and destination IP to the Ethernet MAC module. After completing the transmission of the IP header, the Ethernet TCP/IP module receives and transmits the TCP header and associated data—stored in the SRAM—to the Ethernet MAC module. Fig. 13 depicts the received TCP header and data in the Ethernet MAC module. Finally, a 4-byte CRC is calculated by the Ethernet MAC module, and the data transmission is completed by setting TX_EN to low (0).

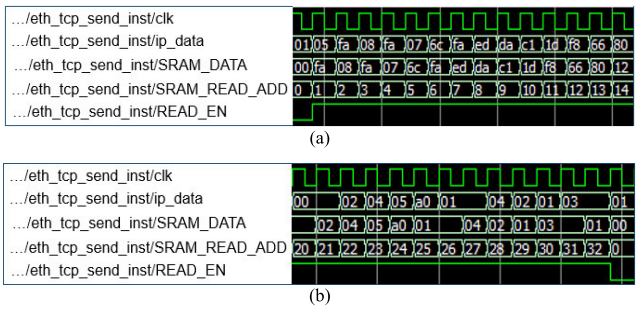

FIGURE 13. Ethernet MAC module receives TCP header and data. (a) Start; (b) stop.

C. FUNCTIONAL VALIDATION WITH FPGA BOARD

Fig. 14 illustrates the system architecture of the data transfer bridge and its corresponding validation environment. The data transfer bridge includes two computers—a desktop and

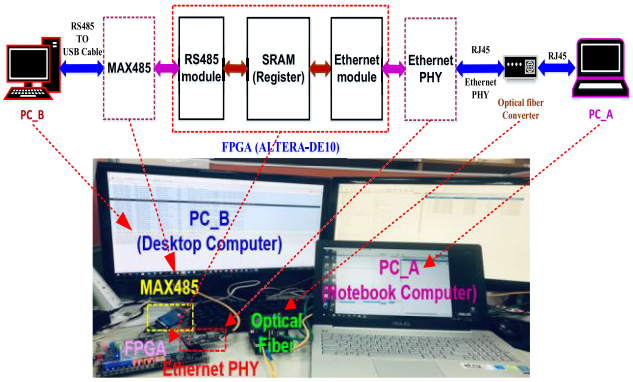

FIGURE 14. System architecture of data transfer bridge and its validation environment.

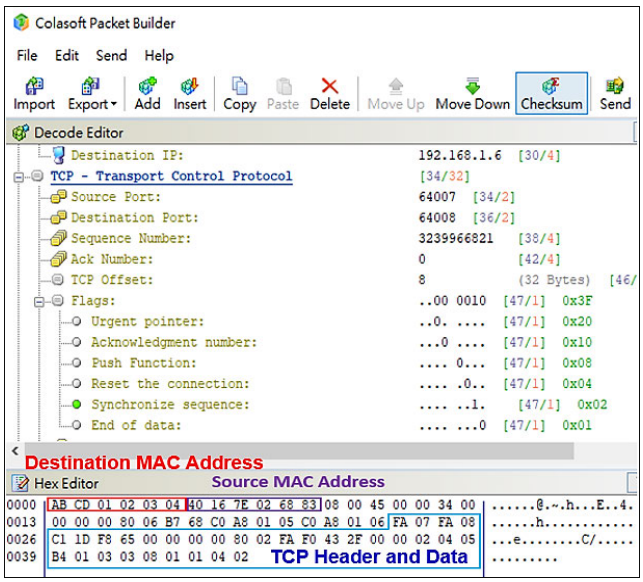

FIGURE 15. Ethernet packet received from notebook computer (PC_A) with Colasoft Packet Builder software.

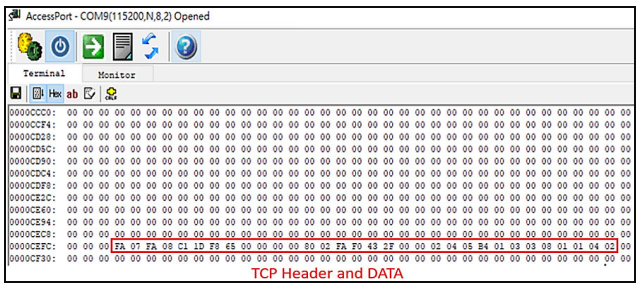

FIGURE 16. TCP header and data received from RS485 via USB cable on the desktop computer (PC_B) with AccessPort software.

a notebook, an optical fiber converter, an Ethernet PHY, an MAX485 transceiver for RS485, and an FPGA development board (DE-10 Standard) [16]. The FPGA board incorporates three custom-designed modules: the Ethernet module, the RS485 module, and the SRAM module (Register). Initially, Host B (PC_B) transmits the RS485 data to

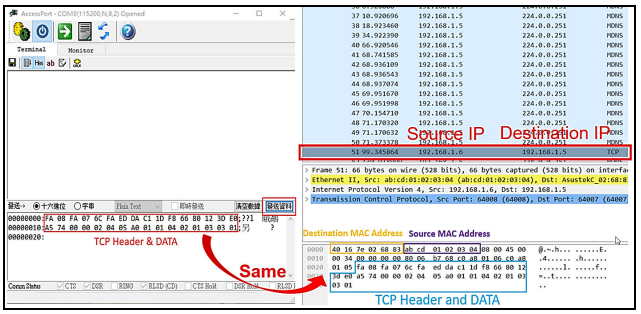

FIGURE 17. Transmitted TCP header and data on a desktop computer with RS485 signal monitoring software (AccessPort) and a notebook computer with packet monitoring software (Wireshark).

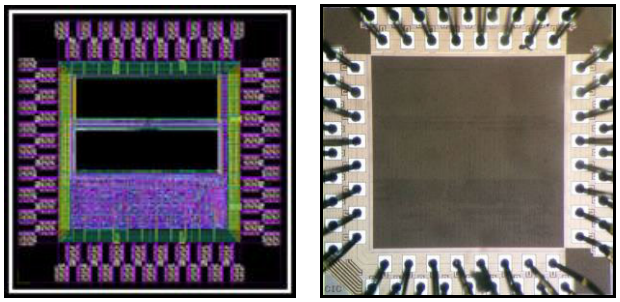

FIGURE 18. ASIC of the Ethernet−−RS485 data transfer bridge with TCP and SRAM. (a) Chip layout; (b) chip photograph.

the FPGA development board (DE-10) through an RS485-to-USB cable and a MAX485 transceiver. The FPGA board (DE-10) receives these data, assembles them into a complete Ethernet packet, and stores the packet in the SRAM (Register). Next, the assembled Ethernet packet is transmitted to Host A (PC_A) through the Ethernet PHY, an RJ45 network cable, and an optical fiber converter. Conversely, Host A (PC_A) transmits an Ethernet packet to the FPGA board (DE-10) through an RJ45 cable and the Ethernet PHY. The on-board Ethernet and RS485 modules disassemble the received Ethernet packet and convert it into RS485 format data. The resulting serial TCP data are then transmitted to Host B (PC_B) through an MAX485 transceiver and an RS485-to-USB cable. Notably, Ethernet packets are parallel data structures, whereas RS485 data are serial. In the verification environment, Host A (PC_A) uses a notebook computer, and Host B (PC_B) uses a desktop computer. The length of the optical fiber in this configuration is approximately 5 km.

Fig. 15 presents an Ethernet packet generated by the Colasoft Packet Builder software on Host A's notebook computer (PC_A). The MAC address of the notebook computer is 40:16:7E:02:68:83 (source), and the IP address of the notebook computer is 192.168.1.5. The destination MAC address (RS485 terminal device) is AB:CD:01:02:03:04, and the IP address is 192.168.1.6. Following packet generation, the Ethernet PHY module receives this Ethernet packet and forwards it to the FPGA development board (DE-10). Upon successful

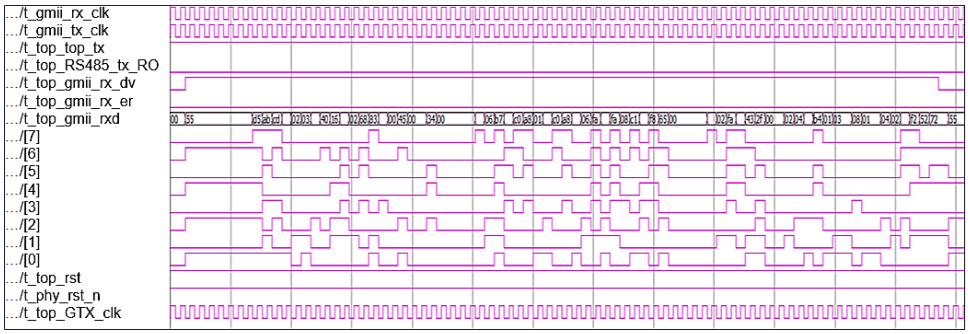

FIGURE 19. Post-layout simulated waveforms transmitted from GMII module to RS485 TX module.

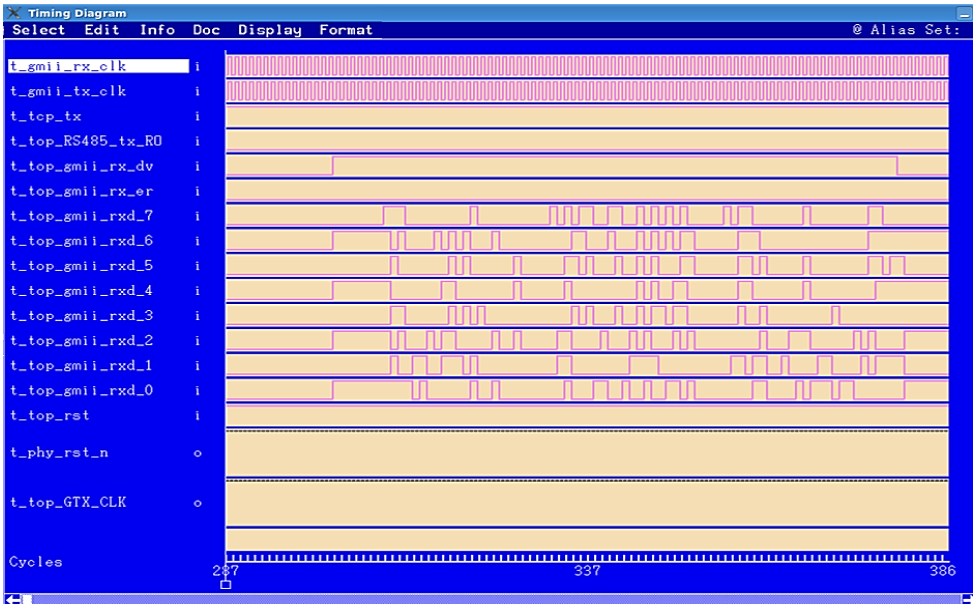

FIGURE 20. Measured waveforms of designed bridge ASIC, which are transmitted from GMII module to RS485 TX module.

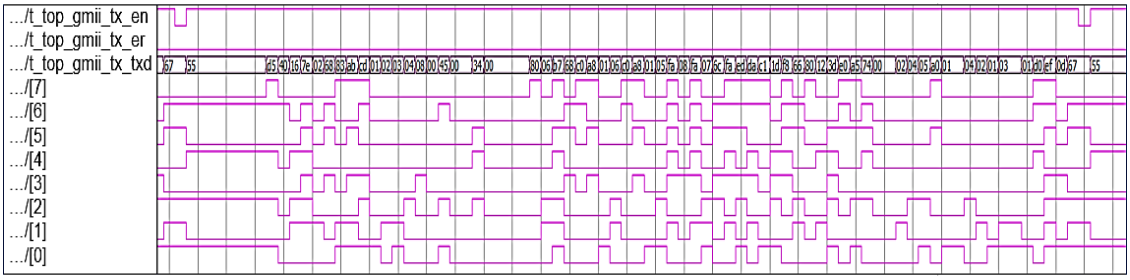

FIGURE 21. Post-layout simulated waveforms transmitted from RS485 TX module to GMII module.

conversion from the Ethernet module to RS485 modules, the RS485 TCP header and data are sent to the MAX485 transceiver. The TCP header and data are then received by the desktop computer (PC_A) using an RS485-to-USB cable. Fig. 16 illustrates the TCP header and data received by Host B's desktop computer using the RS485-to-USB cable with AccessPort software (RS485 signal monitoring software). The received TCP header and data match those transmitted by Host A's notebook computer, confirming the accuracy of the transmission and conversion processes.

Fig. 17 illustrates the TCP header and data as transmitted and monitored on a desktop computer (PC_B) using RS485 signal monitoring software (AccessPort) and on a notebook computer employing packet monitoring software

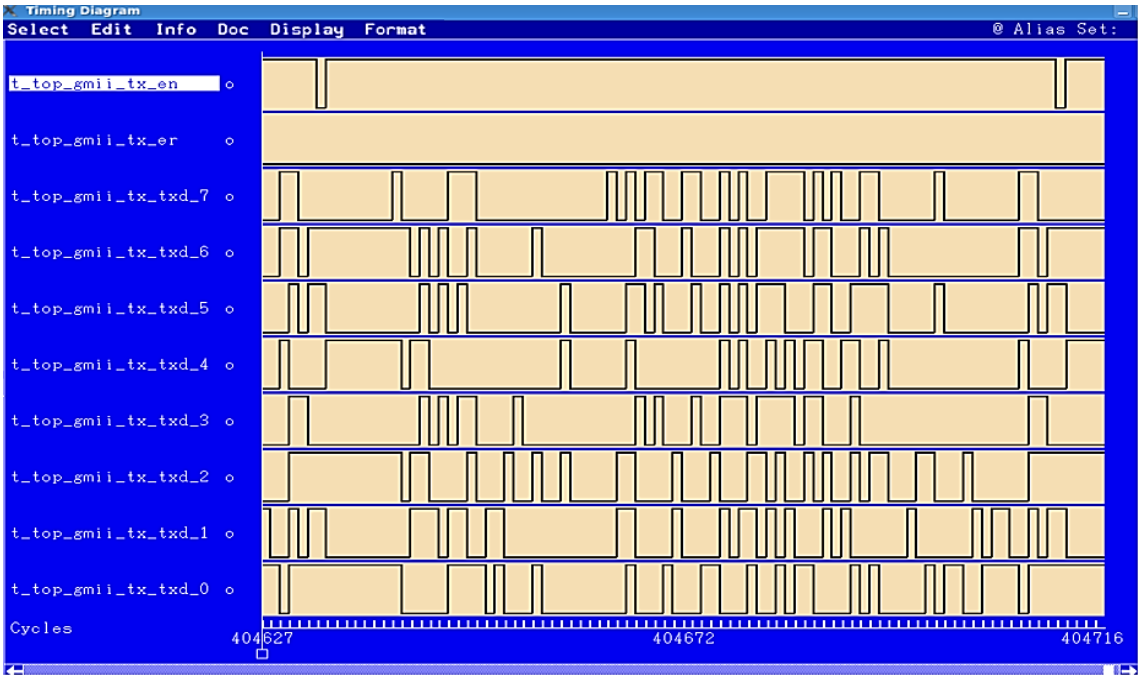

FIGURE 22. Measured waveforms of designed bridge ASIC, which are transmitted from RS485 to GMII module.

(Wireshark). When RS485 data transmission is initiated by a click of the "Send Data" icon, the RS485 module (DE-10) receives the data through an RS485-to-USB cable and an MAX485 transceiver. The transmitted RS485 data are a 32-byte serial packet. Upon passing through the FPGA development board, a complete packet is generated and displayed on the notebook computer running packet monitoring software (Wireshark). This display occurs through the Ethernet PHY and an RJ45 network cable. The destination MAC address is 40:16:7E:02:68:83, and the destination IP address is 192.168.1.5. The source MAC address is AB:CD:01:02:03:04, and the source IP address is 192.168.1.6. The protocol governing this data transfer is TCP. A comparative analysis of the TCP header and data monitored in AccessPort (PC_B) and those in Wireshark (PC_A) confirms the accuracy of data transmission and conversion processes between the desktop (PC_B) and notebook computers (PC_A).

IV. ASIC IMPLEMENTATION AND MEASURED RESULTS OF THE PROPOSED BRIDGE SYSTEM

We evaluated the performance of the bridge ASIC, incorporating SRAM for protocol conversion and data transmission. For the ASIC, debugging and verification were conducted using the NC-Verilog simulator and Verdi/nWave waveform viewer. After the functional verification of the bridge ASIC, we synthesized the bridge ASIC at a clock frequency of 125 MHz by using a TSMC 0.18-μm CMOS cell-based process. The Synopsys IC Compiler was used to plan the chip layout of the Ethernet-to-RS485 data transfer bridge ASIC. After the ASIC passed the design rule check and layout versus schematic verifications, the chip was imaged, as shown in Fig. 18.

TABLE 1. ASIC specifications of the proposed bridge system.

Specification	Values
Technology	TSMC 0.18-μm CMOS
Supply voltage (V)	+ 1.8 V
SRAM (bytes)	256 × 2
Chip size (mm^2)	1.19 × 1.19
Operation frequency (MHz)	125
Baud rate (bps)	115,200
Power consumption (mW)	134.79
Gate count	69,287
Package	CLCC 68

A complete packet was transmitted from the GMII module to the RS485 TX module. When the driving signal of Ethernet RX module (RX_DV) is set to high (1), the parallel data RXD [7:0] are received in sequence. If the parallel data are fully collected, the driving signal RX_DV transitions to low (0), terminating the transmission function. Fig. 19 shows the post-layout simulated waveforms transmitted from the GMII module to the RS485 TX module, captured using the NC-Verilog simulator and Verdi/nWave waveform viewer. Fig. 20 depicts the measured waveforms of the designed bridge ASIC, which were transmitted from the GMII module to the RS485 module and measured with the SoC/SiP ATE Tester (PS1600) from Advantest Corporation. The measured waveforms matched the simulated waveforms.

TABLE 2. Performance comparisons of the proposed bridge ASIC with that of other ethernet and RS485 communication systems.

Parameters	This study	[17] (2013)	[18] (2015)	[19] (2017)	[20] (2018)	[21] (2021)	[22] (2022)
Development process	ASIC	FPGA	FPGA	FPGA	FPGA (ASIC)	DSP	Server Laptop
Operating frequency (MHz)	125	25	125	50	125	–	200
Ethernet (Mbps)	1,000	100	1,000	1,000	1,000	1,000	940
FIFO (bytes)	–	1,500	9,000	–	1,000	–	–
SRAM (bytes)	256 × 2	–	–	2,000	–	16 M	–
Processing latency (μs)	139.31	–	–	–	–	–	–
Throughput (Mbps)	844.88	100	912.8	795.8	–	49.152	50~400
Protocols	IP/TCP	IP/UDP	IP/UDP /ICMP/ARP /PTP	IP/TCP /UDP/ARP /ICMP	IP/UDP	IP/TCP	MULTI TCP/UDP QUIC MPTCP
ASIC chip area (mm^2)	1.19 × 1.19	–	–	–	1.27 × 1.27	–	–
Power consumption (mW)	134.79	–	–	–	137	–	–

Conversely, serial RS485 data were transmitted from the RS485 RX module to the GMII module. When the enable signal of the Ethernet TX module (TX_EN) changes from low (0) to high (1), the parallel Ethernet data TXD [7:0] are transmitted to the GMII module. If the parallel Ethernet data are fully transmitted, the enable signal TX_EN is set to low (0) to end the transmission. Fig. 21 displays the post-layout simulated waveforms transmitted from the RS485 TX module to the GMII module, captured using the NC-Verilog simulator and Verdi/nWave waveform viewer. Fig. 22 illustrates the measured waveforms of the bridge ASIC transmitted from the RS485 to the GMII module. The measured waveforms were captured using the Advantest SoC/SiP ATE Tester (PS1600). The measured waveforms are consistent with the simulated waveforms. The chip layout was correctly fabricated as indicated by the waveforms. Table 1 presents the specifications of the bridge ASIC.

Table 2 summarises the performance of the proposed bridge ASIC with SRAM and compares it with the performance of other Ethernet and RS485 communication systems. Operating at a frequency of 125 MHz, the bridge ASIC addresses the demand for gigabit Ethernet. The design incorporates 256 bytes of on-chip SRAM to reconcile speed disparities between Ethernet and RS485 modules. The maximum throughput of 844.88 Mbps is sufficient for transmitting a 20-byte TCP header and 236-byte data in a propagation time of 2,424 ns. As indicated in Table 2, this throughput achieved with DSP and server-based (PC or workstation) development processes is lower than that achieved with FPGA and ASIC [21], [22]. The processing latency of this study is 139.31 μs, which is the lowest among the studies listed in Table 2. This is because the proposed bridge ASIC performs with a small delay. The proposed ASIC-based solution may be regarded as a substantial contribution. Moreover, the 256-byte on-chip SRAM is optimized for maximum data transfer. Unlike the referenced studies, which did not integrate their proposed architectures into a digital chip, this study successfully implemented the architecture in an ASIC and made sure that it worked correctly. The proposed ASIC not only exhibits low power consumption and a small chip area but also performs with low processing latency and high throughput.

V. CONCLUSION

This study designed and implemented a bridge ASIC for Ethernet packet to RS485 data transfer, incorporating TCP and SRAM. Modules were designed with the Verilog hardware description language and verified on an FPGA development board (DE-10 Standard). After the functional verification was completed with the FPGA development board, the bridge ASIC was synthesized and fabricated using the TSMC 0.18-μm CMOS cell-based process. To achieve gigabit Ethernet functionality at an operating frequency of 125 MHz, the on-chip SRAM was set to 256 kB, enabling successful data transmission up to a maximum of 256 kB and compatibility with the RS485 module operating at a baud rate of 115,200 bps. The processing latency of 139.31 μs and the throughput of 844.88 Mbps are surpassed those of the studies referenced in Table 2, (except for [18]). Unlike the referenced studies, this study integrated the proposed architecture into an actual ASIC. The throughput of the proposed bridge ASIC is higher than that of the referenced studies developed with DSP or a server (PC or workstation). The proposed ASIC exhibited low power consumption, registering at 134.79 mW, and occupied a small chip area of 1.19 × 1.19 mm^2. These advantages—low power consumption, high throughput, small

size, and reduced latency—enhance the industrial applicability of the proposed bridge ASIC. In future work, recognizing and distinguishing between useful and duplicated packets will be important issues.

ACKNOWLEDGMENT

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

REFERENCES

[1] M. R. Caldieri, J. A. Bigheti, and E. P. Godoy, "Implementation and evaluation of wireless networked control systems using modbus," *IEEE Latin Amer. Trans.*, vol. 15, no. 2, pp. 206–212, Feb. 2017.

[2] J. Stój, "Cost-effective hot-standby redundancy with synchronization using EtherCAT and real-time Ethernet protocols," *IEEE Trans. Autom. Sci. Eng.*, vol. 18, no. 4, pp. 2035–2047, Oct. 2021.

[3] L. Lachello, P. Wratil, A. Meindl, S. Schonegger, B. S. Kanunakaran, H. Song, and S. Potier, "System comparison: The 5 major technologies," *Industrial Ethernet Facts*, vol. 2, pp. 1–36, Feb. 2013.

[4] H. Manjule, S. Upadhya, N. Wankhede, M. Kumbhare, K. Thakur, and R. Krishnan, "Ethernet implementation on FPGA," in *Proc. Int. Conf. Emerg. Technol. (INCET)*, Belgaum, India, Jun. 2020, pp. 1–4.

[5] C. H. Yan, Y. Zhou, and S. Du, "RS485 communication protocol based on embedded Linux," *Comput. Sci.*, vol. 34, no. 11, pp. 278–280, Jun. 2008.

[6] B. Jie, "Designing of a communication program with PC as a master in the half-duplex RS485 Netware in Windows," *Nucl. Electron. Detection Technol.*, vol. 24, no. 1, pp. 84–86, 2004.

[7] H.-J. Jia and Z.-H. Guo, "Research on the technology of RS485 over Ethernet," in *Proc. Int. Conf. E-Product E-Service E-Entertainment*, Henan, China, Nov. 2010, pp. 1–3.

[8] L. Ding, P. Kang, W. Yin, and Z.-H. Feng, "Design and implementation of hardware-based low latency TCP offload engine for 10 Gbps Ethernet," in *Proc. 13th IEEE Int. Conf. Solid-State Integr. Circuit Technol. (ICSICT)*, Hangzhou, China, Oct. 2016, pp. 701–703.

[9] L. Ding, P. Kang, W. Yin, and L. Wang, "Hardware TCP offload engine based on 10-Gbps Ethernet for low-latency network communication," in *Proc. Int. Conf. Field-Programmable Technol. (FPT)*, Xi'an, China, Dec. 2016, pp. 269–272.

[10] A. Choudhary, D. Porwal, and A. Parmar, "FPGA based solution for Ethernet controller as alternative for TCP/UDP software stack," in *Proc. 6th, Ed., Int. Conf. Wireless Netw. Embedded Syst. (WECON)*, Rajpura, India, Nov. 2018, pp. 63–66.

[11] G.-M. Sung, Z.-Y. Tan, C.-Y. Lee, C.-L. Tseng, C.-R. Chen, C.-P. Yu, C.-C. Hsiao, and R.-G. Lee, "Ethernet packet transformation and transmission between Modbus/TCP and USB 3.0 with field-programmable gate array development board," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Prague, Czech Republic, Oct. 2022, pp. 1677–1681.

[12] G.-M. Sung, C.-T. Lee, Z.-Y. Yan, and C.-P. Yu, "Ethernet packet to USB data transfer bridge ASIC with modbus transmission control protocol based on FPGA development kit," *Electronics*, vol. 11, no. 20, p. 3269, Oct. 2022.

[13] *IEEE Standard for Ethernet*, IEEE Standard 802.3-2018, Aug. 2018, doi: 10.1109/IEEESTD.2018.8457469. [Online]. Available: https://ieeexplore.ieee.org/servlet/opac?punumber=8457467

[14] G. R. Friedrich and G. H. Reggiani, "Data communication for low resources IoT devices: RS485 over electrical wires," *IEEE Embedded Syst. Lett.*, vol. 16, no. 1, pp. 53–56, Mar. 2024.

[15] *Gigabit Ethernet Transceiver With GMII/MII Support*. Accessed: Feb. 13, 2024. [Online]. Available: https://www.mouser.com/pdfdocs/KSZ9031MNX.pdf

[16] *DE10-Standard Board: Cyclone V*. Terasic Headquarter, Hsinchu, Taiwan. Accessed: 13, Feb. 2024. [Online]. Available: https://www.terasic.com.tw/cgi-bin/page/archive.pl?

[17] S. Nayeema and K. Jamal, "Design and implementation of MAC transmitter for the transmission of UDP packet using FSM and Verilog coding techniques," *Int. J. Eng. Res. Appl.*, vol. 3, no. 1, pp. 338–342, Feb. 2013.

[18] P. Födisch, B. Lange, J. Sandmann, A. Büchner, W. Enghardt, and P. Kaever, "A synchronous gigabit Ethernet protocol stack for high-throughput UDP/IP applications," *J. Instrum.*, vol. 11, no. 1, Jan. 2016, Art. no. P01010.

[19] Q. Liu, Z. Xu, and Z. Li, "Implementation of hardware TCP/IP stack for DAQ systems with flexible data channel," *Electron. Lett.*, vol. 53, no. 8, pp. 530–532, Apr. 2017.

[20] H.-K. Wang, C.-P. Yu, G.-M. Sung, and M.-W. Li, "Intelligent packet transformation and transmission between Ethernet and optical fiber systems based on a field-programmable gate array board," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Miyazaki, Japan, Oct. 2018, pp. 4071–4076.

[21] N. Pekez, A. Popovic, and J. Kovacevic, "Ethernet TCP/IP-based audio interface for DSP system verification," *IEEE Consum. Electron. Mag.*, vol. 10, no. 1, pp. 45–50, Jan. 2021.

[22] S. Hätönen, A. Rao, and S. Tarkoma, "Programmable session layer MULTI-connectivity," *IEEE Access*, vol. 10, pp. 5736–5752, 2022.

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 sensors, USB/RS485 communication, analog-to-digital converter IC, motor control IC, radio-frequency harvester IC, AI chip design, and mixed-mode ICs for use by the Internet of Things (IoT).

CHUN-TING LEE was born in Taiwan, in 1977. He received the M.S. degree in electronics engineering from National Tsing Hua University. He is currently pursuing the Ph.D. degree. Since 2005, he has been an IC Designer. His research interests include magnetic transducers, analog front-end ICs, mixed-mode ICs, and power manager ICs for vehicles.

ZHI-LUN YOU was born in Taiwan, in 1995. He received the B.S. degree in electrical engineering from Tamkang University, New Taipei, Taiwan, in 2017, and the M.S. degree in electrical engineering from the National Taipei University of Technology, Taipei, Taiwan, in 2019. Since 2017, he has been an IC Designer. His research interests include digital ICs and Ethernet ICs.

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, USA, in 1995. He is currently an Instructor with the Department of Electrical Engineering, National Taipei University of Technology. His research interests include magnetic sensor applications, analog filter circuits, and mixed-mode circuit design.