Scalable Hardware Mechanism for Partitioned Circuits Operation

Yusuke Katoh[1], **Hironari Yoshiuchi**[2], **Yoshio Murata**[3], and **Hironori Nakajo**[4]

ABSTRACT

For designing hardware with a high-level synthesis tool using a programming language such as C or Java, its large size of logic circuit makes it difficult to implement the design in a single FPGA. In such a case, partitioning the logic circuit and implementing in multiple FPGAs is a commonly used approach.

We propose the Scalable Hardware Mechanism, which enables the operation of a partitioned circuit to prevent the degradation of clock frequency by minimizing its dependence on the usage and the type of FPGA. Our mechanism provides a reduced delay by the collective signal transmission with the partitioned AES code generation circuit and the character string edit distance calculation circuit as partitioned circuits. The collective signal transmission has attained 1.27 times improvement in the speed for the AES code generation circuit and 3.16 times improvement for the character string edit distance calculation circuit compared with the circuit by the conventional method.

Keywords: Circuit operation, Partitioned circuit, FPGA

1. INTRODUCTION

FPGAs have often been used in network devices, embedded systems and as prototypes in ASIC design. In recent years, due to larger integration and speedup of FPGAs, high-performance computing expects hardware acceleration with them. As increasing scale of hardware application and utilizing high-level synthesis (HLS) tools require a large amount of logic, it is difficult to implement such logic into a single FPGA. Therefore, in recent years there have been attempting to partition a large-scale circuit into multiple sub-circuits and implement them into multiple FPGAs.

Conventional operation for partitioned circuits with multiple FPGAs has been conducted with synchronization under a global clock, which is called an emulation clock for circuit emulation. However, partitioning a circuit may cause transmission delay among the divided circuits due to a global clock, which causes degradation of the operating frequency of the system, thereby reducing the performance of the whole system.

In this research, we introduce a new partitioned circuit operation mechanism called Scalable Hardware Mechanism, which corresponds to some kinds of circuit operations as well as circuit emulation using multiple FPGAs.

In a Scalable Hardware Mechanism, a large circuit is partitioned into a sending side and receiving one. Packetized signal information is sent from a sending side together to a receiving side to enhance operation speed in a whole circuit under a dedicated protocol. Since the protocol does not depend on a specific communication interface, the operating circuit keeps scalability as well as can be configured to a massive system consisting of a large number of FPGAs.

Moreover, the method prevents degradation of the circuit operating speed due to global synchronization to improve the operation speed of the whole circuit.

Our Scalable Hardware Mechanism cannot be applied to all kinds of partitioned circuits. Applicable circuits are limited to one-way signals between the partitioned circuits because backward signal may cause a change in the state of the previous-stage finite state machine. However, for partitioned circuits connected with one-way signals, such as stream application, the Scalable Hardware Mechanism can improve the speed of operation of the partitioned circuits. As increased the number of ports to connect FPGA boards, partitioned circuits can be configured not only in a one-dimensional array but star or tree topology in which the direction of signal is limited in one-way.

2. RELATED WORKS

In implementing divided circuits into multiple FPGAs using two or more boards and synchronizing the whole divided circuits, a global common clock, called emulation clock, is often used. The period of this emulation clock is generally estimated from the delay time between the registers for all the signal wires. In some research, for the large number of signal transmissions, high-speed serial communication, instead of

Manuscript received on August 27, 2018 ; revised on November 2, 2018.
Final manuscript received on December 2, 2018.
[1] TOSHIBA INFORMATION SYSTEMS (Japan) CORPORATION, Japan
[2] NTT Advanced Technology, Japan
[3,4] Institute of Engineering, Tokyo University of Agriculture and Technology, Koganei, Tokyo 184–8588, Japan Email: nakajo@cc.tuat.ac.jp

bus connection, is sometimes used to overcome the pin-neck of FPGAs. With the high-speed serial communication, degradation of the emulation clock is also avoided to secure the clock frequency of the whole circuit[3][4].

However, when an emulation clock is used, clock frequency may degrade due to increased transmission delay as the number of ports in the FPGA is increased.

Moreover, an emulation clock cycle needs to be calculated in the worst case in which the largest transmission delay is supposed in implementing circuits into FPGAs in each design. Therefore transmission method via an unstable transmission time such as Ethernet is difficult to be used in such a emulation clock based system. Some large-scale multi-FPGA systems with a large amount number of high-end FPGAs adopt high-speed serial communication for high performance computing[1]. Moreover, a large-scale FPGA system which adopts 512 of low-end FPGAs, Xilinx Spartan3, to build a huge systolic array has been also implemented and evaluated with some practical applications[2].

A set of constraints during the partitioning task and an iterative routing algorithm to obtain the best multiplexing ratio for inter-FPGA signals have been researched[5]. This research focuses on a method to partition a circuit efficiently but our approach focuses on the way to operate the partitioned circuits in high speed.

3. SCALABLE HARDWARE MECHANISM

3.1 Concept of the Scalable Hardware Mechanism

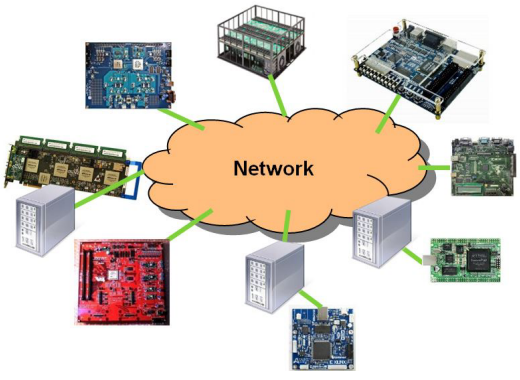

Fig.1: *Overall concept of Scalable Hardware Mechanism*

In the Scalable Hardware Mechanism, a circuit is partitioned and implemented into multiple FPGAs with which multiple boards are connected via a network to realize a virtual large-scale circuit as shown in Fig.1. Our proposed method has the following advantages compared with the conventional method.

- **Scalability**: The conventional method is conducted in an operation environment with the specified FPGA cards in the partitioned circuits. However, the Scalable Hardware Mechanism does not depend on the board configurations or on the type or grade of the FPGA. In our system, FPGA boards can be connected via different network interfaces such as PCI Express, Gigabit Ethernet or RocketIO to make the partitioned circuits more scalable.
- **Asynchronicity among partitioned circuits**: To operate the partitioned circuits with a local clock, conventionally all circuits should be synchronized with the clock.

Also, Imperial College London developed CUBE, which has 64 FPGA devices on a single board and 512 FPGA devices, as a platform providing simple interface and streamline processing power. While CUBE is dedicated to stream applications with connecting multiple FPGA boards in a single-dimensional array, a clock signal is supplied from the FPGA in the previous stage board in the array.

In the Scalable Hardware Mechanism, since each FPGA is operated under synchronization of the local clock, each partitioned circuit can be operated asynchronously in higher speed. Therefore, the mechanism restrains degradation of clock frequency of a circuit and maximizes the operating speed with the Hardware Expansion Protocol.

3.2 Hardware Expansion Protocol

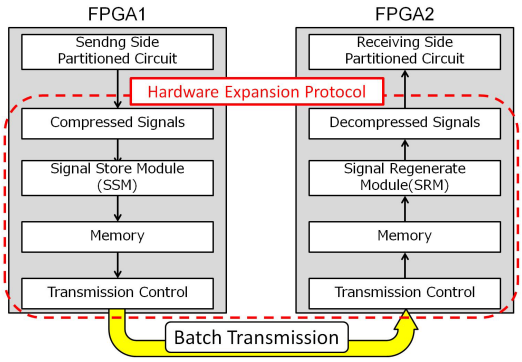

Fig.2: *Processing flow of signal information*

As shown in Fig.2, a protocol called the Hardware Expansion Protocol realizes an efficient operation of the partitioned circuits for a Scalable Hardware Mechanism using the Transmission Control Method, which makes the transmission independent of the communication interface.

In a Hardware Expansion Protocol, after a bundle of signal information in tens to hundreds of clock cycles are stored into memory, the bundled information is sent to the next FPGA with collective signal transmission. With all these mechanisms, each partitioned circuit is able to operate in each FPGA independent of the size of the targeted circuit, the type of the board or an FPGA, and the distance between

the boards.

3.2.1 Signal Storing and Regenerating Mechanisms

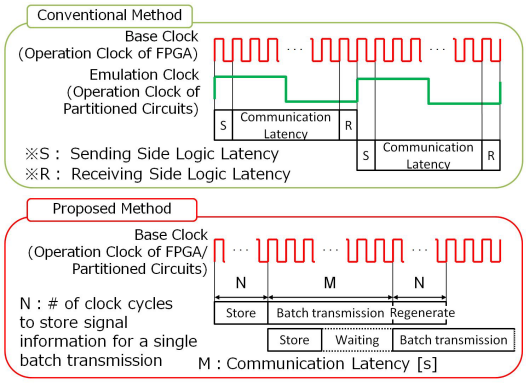

Fig.3: *Latency hiding with collective transmission of bundle of signal information*

In execution control of the conventional partitioned circuits, in order for all partitioned circuits to synchronize, all the signal information for one clock should be outputted from each partitioned circuit by the time of the start of the following clock to keep coordination among all the circuits. Therefore, it is necessary to include propagation delay of circuits in both sending and receiving sides as well as communication delay between boards within a single cycle of the emulation clock.

On the other hand, after signal information of tens or hundreds of clock cycles outputted from the partitioned circuit is stored into memory, the bundle of signal information is transmitted to the next circuit. Then, after the sent signal information is received by the receiving circuit, the signals are regenerated and operated one by one clock cycle by the circuit.

Since the operation clock in each FPGA is given as the original clock of the FPGA, continuous communication between partitioned circuits can be overlapped with each operation in each circuit. This effectively overcomes the transmission delay between the circuits.

Moreover, since the proposed mechanism can work independently on different network interfaces or communication mediums, maximum payloads of various transmission systems can be utilized, which prevents the throughput degradation.

Signal Storing and Regenerating Mechanisms are realized by implementing Signal Storing (SSM) and Signal Regenerating Module (SRM) respectively. An SSM controls a circuit on the sending side and a buffer memory that stores signal information for collective transmission. It always observes the buffer memory status, and prevents it from overflowing. If the buffer memory become almost full, the SSM stops supplying the clock signal to the circuit on the sending side. After more room is available in the buffer, the SSM restarts supplying the clock to store the bundle of signal information.

The SRM controls circuits on the receiving side as well, and the buffer memory that stores sent signal information from the source circuit. In the buffer control, when the SSM checks the status of the receiving buffer which is empty or not, it sends signal information one by one clock cycle from the buffer memory to the partitioned circuit in the case that there exists signal information in the buffer. When the buffer memory is empty, the SRM stop supplying clock signal to the circuit in the receiving side. After the signal information will be stored into the buffer, the SRM restart supplying the clock to operate the circuit in the receiving side.

3.2.2 Transmission Control

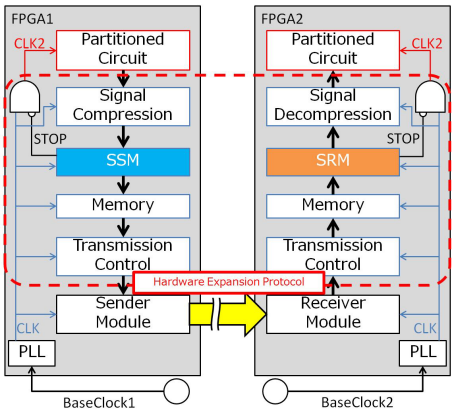

Fig.4: *Dataflow between partitioned circuits*

By packetizing signal information, independent communication on a communication medium among partitioned circuits can be realized. However, in order to establish a communication path between the boards in which partitioned circuits are implemented, the system requires another communication control between FPGAs.

The proposed system adopts a control packet, which checks the status of the buffer memory for communication and a data packet that includes the output signal information on a partitioned circuit.

To establish communication, the SSM on the sending side sends a control packet including a Request signal. If the receiving buffer is not almost full, and can receive a packet, the SRM on the receiving side sends back a control packet including an Ack signal to establish communication. Otherwise the SRM sends a Nack signal to the sending side to postpone sending. After establishing the communication, when the sending buffer becomes almost full, the SRM sends a packet including an Xoff signal to stop sending packets. Then the buffer reserves room for receiving packets later, and the SRM sends an Xon signal to the sending side to restart sending the packet.

Using the Ack and the Nack signals, which are

included in the control packet, we can realize communication establishment, and the Xon / Xoff signal can realize the flow of control between partitioned circuits. The data flow coupling all the mechanisms of the proposed architecture is shown in Fig.4.

3.3 Implementing Partitioned Circuits

A designed large-scaled Verilog HDL source code is transformed into an abstract syntax tree using Pyverilog[6] which is an analyzing tool for Verilog HDL. A transformed syntax tree is input to a developed circuit partitioning tool to generate multiple partitioned Verilog HDL source codes.

As mentioned before, in a Scalable Hardware Mechanism, a direction of signals is limited to a single way. Thus, in order for partitioned circuits to be operated in the Scalable Hardware Mechanism, a direction of signals between the partitioned circuits should be limited to a single way.

4. EVALUATION

4.1 Evaluation Environment

Table 1: *Implementation and evaluation environment*

Used HDL	Verilog HDL
Evaluated device	Virtex-5 XC5VLX330-1FF1760
Design tool	ISE Design Suite 14.6
RTL simulation tool	ISim 14.6

The implementation and evaluation environment is shown in Table 1.

4.2 Target Application Circuit

4.2.1 AES Code Generation Circuit

AES is a common key encryption system which divides a plain sentence into 128-bit blocks for encryption using the encryption key of the key length of 128,192,256 bits. Encryption applies the following operations on the input data.

(1) SubBytes: Change the data of one Byte into the value of another Byte using the character conversion table.
(2) ShiftRows: Replace the position of data per Byte.
(3) MixColomns: Change the data of 4Byte by a bit operation.
(4) AddRoundKey: Apply an XOR operation with a round key.

Steps from (1) to (4) are repeated in the specified round number to encrypt the target. In the repeat, output of the step (4) is input to the step (1).

The AES code generation circuit, which is used for evaluation, treats the key length of 128 bits to generate an AES code in the ECB mode. An AES code generation circuit can be divided into a key generation circuit and a code generation circuit.

In a key generation circuit, after ten round keys are generated from a single key, they are transmitted to a code generation circuit. In a code generation circuit, the information on the round key transmitted from the key generation circuit is used to execute scrambling of a plain sentence. Since a key generation circuit transmits information of a 128-bit round key, a 1-bit round key output flag and a 1-bit reset signal, altogether 130-bits signal information is transmitted between the partitioned circuits.

4.2.2 Character String Edit Distance Calculation Circuit

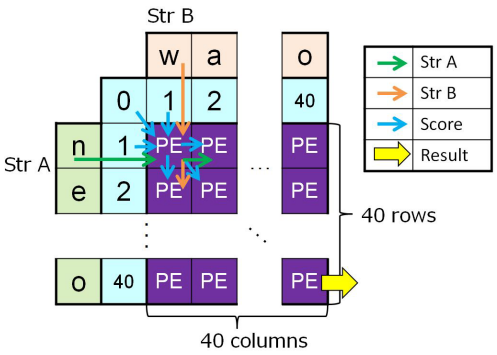

Fig.5: *Concept of character string edit distance calculation circuit*

A character-string edit-distance calculation circuit computes the degree of similarity of a character string by measuring the edit distance (Loewenstein distance) of two-character strings. Edit distance is the number of deletions, insertions and substitutions of characters needed to change a given character string to the targeted character string. A low Loewenstein distance means highly similar character strings.

The calculation algorithm of the Loewenstein distance is as follows. We let the number of characters of the character string A and string B be lenA and lenB with the number of characters as i($1 \leq i \leq lenA$) and j($1 \leq j \leq lenB$) respectively. Then the score table ST for calculating the Loewenstein distance is prepared as $(lenA+1) \times (lenB+1)$. Equation 1 is used to calculate the value assigned to the score table.

$$min \begin{cases} ST(i-1,j)+1 \\ ST(i,j-1)+1 \\ ST(i-1,j-1)+a \end{cases} \quad (1)$$

(1) Zero is substituted into ST(0, 0).
(2) i and j are substituted into ST(i, 0) and ST(0, j) respectively.
(3) The i-th character of the string A and the j-th character of the string B is compared. If they are equal, the temporary variable a is set to 0, otherwise it is set to 1.
(4) Calculated value by expression 1 is assigned to ST(i, j).

First, steps (1) and (2) are executed to initialize the score table. Then, steps (3) and (4) are executed repeatedly until the conditions $i = lenA$ and $j = lenB$ become true. The value of ST(lenA, lenB) is finally calculated as the Loewenstein distance. We have implemented and evaluated a circuit which compares two 40-character strings based on the circuit. The comparison circuit of 40 characters consists of 40×40 two dimensional array of a processing element (PE), which performs character-wise comparisons. One PE executes processing shown in equation 1. Therefore, the calculation of the degree of similarity of 40 characters is outputted in 80-cycle latencies in the unpartitioned original circuit.

The partitioned circuits in this evaluation is generated by dividing the original circuit at the point between the 20th and the 21th rows. The width between the divided circuits becomes 1,872 bits as shown in Table 2. Output from each PE uses 42 bits and each character has 8 bits. In the evaluation circuit, 50 bits for control are padded to the 1,872 bits, thus a total of 1,922 bits are transmitted between divided circuits.

Table 2: *The number of transmitted bit between divided circuits*

Item	The number of bit [bits]
Output bits from PE	$42 \times 40 PE = 1680$
Divided character string (20character)	$2 \times 20 = 160$
Output	32
Total	1872bit

4.3 Evaluation Results

For evaluation, we have calculated emulation clock frequency in a conventional method and delay in collective signal transmission of our proposed method in section 4.3.1 and section 4.3.2 respectively. The operation time is measured by RTL simulation with these parameters.

A synchronous circuit with an emulation clock is used as the circuit of the conventional method for comparison. The details of the circuits of the conventional and proposed methods are shown below.

4.3.1 Circuit in the Conventional Method

Operation of the divided circuits of the conventional method is performed by an emulation clock. Since a single cycle of this emulation clock must include critical path delay and the transmission delay between circuits , the cycle of this clock becomes longer than the original clock cycle of an FPGA. This cycle T_{emu} is estimated by equation 2 given in [4]. This equation is estimated with the GTX transceiver, which is a communication module from Xilinx.

$$T_{emu} = T_{cp} + T_{gtx} + T_d + T_f \times N \qquad (2)$$

- T_{emu}: Emulation cycle time [ns]
- T_{cp}: Critical path delay [ns]
- T_{gtx}: GTX Transceiver latency [ns]
- T_d: Latency of received data to output [ns]
- T_f: Transmission time of a frame data [ns]
- N: Count of transmission of frame data

The parameters in this evaluation are set as follows. The clock frequency of the FPGA is required to be 100 MHz. As T_{cp} is required for T_d, because of the synchronization of the base clock and the emulation clock, it becomes the delay of the half cycle of the base clock: T_d is set to be half of 10 ns = 5 ns. T_{gtx} is set to be 49 ns as shown in [4]. When the transmission rate of a GTX transceiver is 5 Gbps, T_f is set to 4ns; and when the transmission rate is 10 Gbps T_f, it is set to 2ns Moreover, 20-bits signal information can be transmitted for a single frame. The signal information transmitted in one clock cycle is 140-bits: 130 bits padded with 10 bits in an AES code circuit. In a character string edit distance calculation circuit, a total of 1,880-bits are transmitted, which includes eight padded bits to the 1,872 information bits.

Based on the above assumption, T_{emu} is set to 92 [ns] when the transmission rate is 5 Gbps, corresponding to the frequency of 10.869 MHz in the AES code generation circuit. When a transmission rate is assumed to be 10 Gbps, T_{emu} would be set to 78 [ns], corresponding to the frequency of 12.829 MHz.

In the character string edit distance calculation circuit, when the transmission rate is 5 Gbps, T_{emu} is assumed to be 440 [ns], corresponding to the frequency of about 2.272 MHz. When the transmission rate is assumed to be 10 Gbps, T_{emu} is 252 [ns], corresponding to the frequency of about 3.968 MHz. By setting the clock at the frequency of operation of the unpartitioned original circuit, calculation of the processing time in the conventional method is attained.

4.3.2 Circuit in the Proposed Method

In the evaluation of the proposed method, the circuit of the Hardware Expansion Protocol as shown in figure 4 is added to the divided circuits. However, in order for this evaluation to estimate reduction of the transmission delay by collective signal transmission, compression / extension circuits are not coupled into the evaluation circuit. Moreover, since the two divided circuits are used for evaluation, transmission control circuit that performs communication control with more than three FPGAs is not coupled. FIFO is implemented as a send and receive buffer. The bit width which can be written in a single clock cycle to the FIFO is 130 bits for the AES code generation circuit and 1,922 bits for the character string edit distance calculation circuit with 1024 depth in the FIFO.

Additionally, a pseudo transmission circuit is inserted instead of a sending and a receiving module. After the pseudo transmission circuit receives output from the FIFO on the sending side, it holds the signal information until the specified time transmission

delay time elapses. The transmission delay T_{delay} is specified in the equation 3.

$$T_{delay} = T_{gtx} + T_{buf} + (C_{bit} + P_{bit}) \times M/R \quad (3)$$

- T_{delay}: Transmission delay of collective transmission [ns]
- T_{gtx}: GTX Transceiver latency [ns]
- T_{buf}: Transmission delay to check a receiving buffer [ns]
- C_{bit}: The number of bits from partitioned circuit per clock [bit]
- P_{bit}: The number of padding corresponding to a frame [bit]
- M: Clock count to store signal information
- R: Transmission rate [bps]

In order to correspond to the conditions of the conventional method, the clock frequency of the FPGA is set to 100 MHz. Moreover, T_{gtx} is assumed to be 49 ns. T_{buf} is calculated by T_{gtx}+ receiving side critical path delay +1 frame transfer time. Since critical path delay is to be 10 ns from the 100 MHz clock frequency of the FPGA. The transfer time of one frame is set to 4 ns when transmission rates are 5 Gbps, and it is set to 2 ns for the transmission rate of 10 Gbps. Signal information per clock C_{bit} is 130 bits in an AES code generation circuit, and is 1,922 bits in a character string edit distance calculation circuit. Since one frame is set to 20 bits, P_{bit} is to be 10 bits in an AES code generation circuit, and is set to be 18 bits in a character string edit distance calculation circuit. Moreover, M is set to 256 in this evaluation. Consequently, $(C_{bit} + P_{bit}) \times M$ is the total number of bits to be transmitted together. When the transmission rate is 5 Gbps, R becomes 5×10^9, and when the transmission rate is 10 Gbps, R becomes 10×10^9.

Based on the above assumption, when the transmission rate is 5 Gbps, T_{delay} is set to 7,275 [ns], and when the transmission rate is 10 Gbps, T_{delay} becomes 3,689 [ns] in the AES code generation circuit.

In the character string edit distance calculation circuit, when the transmission rate is 5 Gbps, T_{delay} is set to 99,377 [ns], and when the transmission rate is 10 Gbps, T_{delay} becomes 49,713 [ns].

4.3.3 Evaluation of Logic Size

The evaluation of logic size is shown in Table 3 and Table 4.

From the result, increasing ratio of LUT and registers is no so significant when the system is coupled with the proposed mechanism. However, the number of BlockRAM increases significantly with the proposed mechanism because the signal information is not compressed yet in this evaluation. An efficient compression algorithm is necessary for the signal information against this problem.

4.3.4 Evaluation of operation time

For the evaluation of operation time, the specific plain sentences and encryption key are given to be performed 10,000 encryption in the AES code generation circuit. In the character string edit distance calculation circuit, the specific input strings are given to be calculated 10,000 times to determine the similarity of character strings. We have measured the time taken to complete these calculations in each circuit. The evaluation results of the operation time are shown in Figures 6 and 8. Moreover, the stalling times of the partitioned circuit in operating time are shown in Fig.7 and Fig.9 for each circuit.

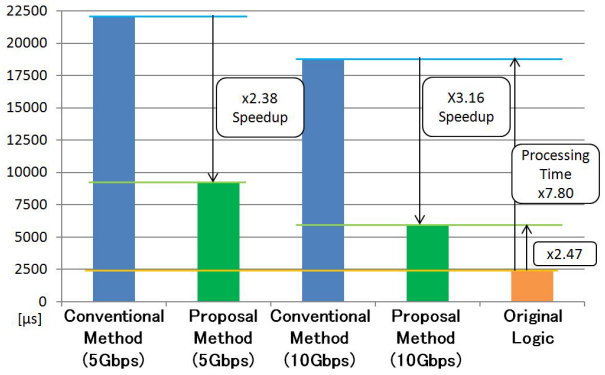

Fig.6: *Operation time (AES code generation circuit)*

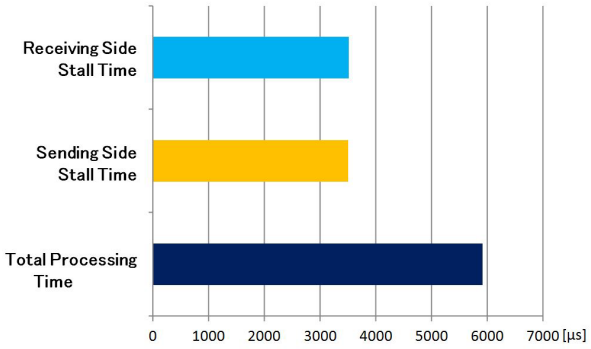

Fig.7: *Stalling time ratio (AES code generation)*

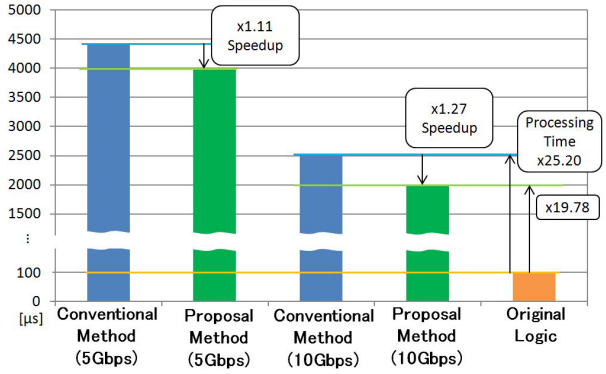

Fig.8: *Operation time (character string edit distance calculation circuit)*

Table 3: *Logic size with an AES code generation circuit*

	Sending side		Receiving side		
	Proposal	Unpartitioned	Proposal	Unpartitioned	The used number
LUT	703(0.3%)	647(0.3%)	1070(0.5%)	1008(0.5%)	207360
Registers	656(0.3%)	457(0.2%)	311(0.1%)	141(0.1%)	207360
BlockRAM	4(1.4%)	0(0%)	4(1.4%)	0(0%)	288

Table 4: *Logic size with a character string edit distance calculation circuit*

	Sending side		Receiving side		
	Proposal	Unpartitioned	Proposal	Unpartitioned	The used number
LUT	143240(69.1%)	143153(69%)	170007(82%)	166009(80.1%)	207360
Registers	50926(24.6%)	48848(23.6%)	67010(32.3%)	51760(25%)	207360
BlockRAM	72(25%)	18(6.3%)	54(18.8%)	0(0%)	288

In the case of the AES code generation circuit, the proposed method improves by 2.38 times compared to the conventional method when the transmission rate is 5 Gbps, and by 3.16 times for 10 Gbps. In the case of the character string edit distance calculation circuit, when the transmission rate is 5 Gbps, the proposed system is faster by 1.11 times; and, for the case of 10 Gbps, 1.27 times. In both instances of the transmission rate between partitioned circuits of 5 Gbps or 10 Gbps, in execution time, improvement from the conventional method technique can be realized.

This is because the latency in the conventional circuit is concealed by transmitting the signal information for 256 clocks continuously. The latencies of the GTX transceiver are largely reduced compared to the circuit of the conventional method. Thus, by transmitting signal information collectively, communication time can be overlapped with the operation of the circuit. In this way, our proposed method is advantageous compared to the emulation clock method.

In the signal storing and the signal regenerating module, latency is implicit in the transmission with storing the signal information to a buffer in parallel with the transmitting signal information. However, in the evaluation of the character string edit distance calculation circuit, this effect has been reduced greatly. From Fig.9, a significantly long stalling time is found in both the sending and the receiving side. Therefore, almost all the time of the whole operation time is consumed in transmitting signal information. The succeeding transmission should wait for the time when the current collective signal transmission is finished. When the current transmission is finished, the next transmission is started immediately. It seems to reduce the effect of the transmission latency implicit in the collective signal transmission.

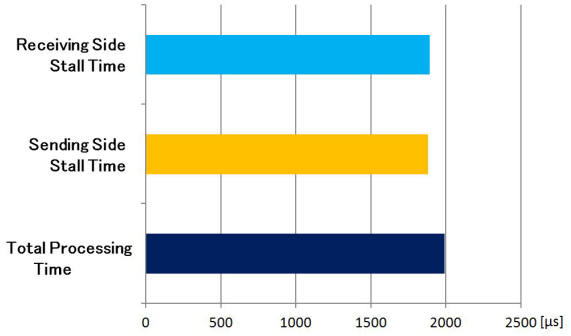

Fig.9: *Stalling time ratio (character string edit distance calculation)*

5. CONCLUSION

In this paper, we have proposed a Scalable Hardware Mechanism in which the collective signal transmission is performed by Signal Storing and Signal Regenerating Modules. In the case of the AES code generation, speedup gained by the proposed method is up to 3.16 times faster than the emulation clock synchronization method. In the case of the character string edit distance calculation, a maximum improvement of 1.27 was achieved. In the future, we plan to implement an efficient compression algorithm to reduce the time to store and transmit signal information between the partitioned circuits.

ACKNOWLEDGMENT

This work was partially supported by JSPS Grant-in-Aid for Scientic Research (C) Number 16K00078.

References

[1] H. Morisita, K. Inakagata, Y. Osana, N. Fujita and H. Amano, "Implementation and evaluation of an arithmetic pipeline on FLOPS-2D: multi-FPGA system," *ACM SIGARCH Computer Architecture News*, Vol.38, No.4, pp.8–13, September 2010.

[2] O. Mencer, K. H. Tsoi, S. Craimer, T. Todman, W. Luk, M. Y. Wong, and P. H. W. Leong, "Cube: A 512-fpga cluster In Programmable Logic," *2009 5th Southern Conference on Programmable Logic (SPL)*, Sao Carlos, pp.51–57, 2009.

[3] K.Takahashi, R.Saeki, M.Kuga and T.Sueyoshi, "Circuit Partitioning Techniques for FPGA-based ASIC Emulator via High-speed Serial Communication," *Proc. 2011 Joint Conference of Electrical and Electronics Engineers in Kyusyu*, 11–1P–03, pp.249-250, 2011.

[4] K.Takahashi, M.Amagasaki, M.Kuga, M.Iida, T.Sueyoshi, "Circuit Partitioning Methods for FPGA-based ASIC Emulator using High-speed Serial Wires," *Proc. The 17th Workshop on Synthesis And System Integration of Mixed Information Technologies(SASIMI2012)*, pp.317–318, 2012.

[5] M. Turki, Z. Marrakchi, H. Mehrez and M. Abid, "Frequency Optimization Objective during System Prototyping on Multi-FPGA Platform," *International Journal of Reconfigurable Computing*, Volume 2013, Article ID 853510, 2013.

[6] Shinya Takamaeda-Yamazaki: Pyverilog: "A Python-based Hardware Design Processing Toolkit for Verilog HDL," *11th International Symposium on Applied Reconfigurable Computing (ARC 2015) (Poster), Lecture Notes in Computer Science*, Vol.9040/2015, pp.451–460, 2015.

Yusuke Katoh received the BE and ME degrees from Tokyo University of Agriculture and Technology Japan, in 2013 and 2015, respectively. He is currently an employee of Toshiba Information Systems Corporation.

Hironari Yoshiuchi received the BE and ME degrees from Tokyo University of Agriculture and Technology Japan in 2014 and 2016, respectively. He is currently an employee of NTT Advanced Technology Corporation, Japan.

Yoshio Murata received the BE degrees from Tokyo University of Agriculture and Technology Japan in 2016. He is currently a student of Graduate School of Tokyo University of Agriculture and Technology. His research interests include circuit partitioning and distributed debug environment.

Hironori Nakajo received the B.E. and M.E. degree in Electrical Engineering from Kobe University in 1985 and 1987, respectively. He visited Center for Supercomputing Research and Development (CSRD) of the University of Illinois at Urbana-Champaign as a Visiting Research Assistant Professor from 1998 to 1999. He is an Associate Professor at Institute of Engineering, Graduate School Tokyo University of Agriculture and Technology since 1999. His research interests are computer architecture, parallel processing, cluster computing and reconfigurable computing. He is a member of IPSJ, IEEE and ACM. Ph.D (Doctor of Engineering).