Proposal of Publish/Subscribe Communication Framework for Circuit Components on FPGA

Kenta Arai[1,a)] Takeshi Ohkawa[1,b)] Kanemitsu Ootsu[1] Takashi Yokota[1]

1. Introduction

Publish/Subscribe communication model [1] is widely recognized as a model to represent data flow effectively. In the model, subscribers receive only the necessary topics among the information sent by publishers. This represents the natural structure of information processing in the era of big data. This model is used in various fields to construct a parallel distributed processing system. For example, it is employed by ROS (Robot Operating System) [2], which is one of the most popular robotics frameworks. ROS improves scalability, expansibility and resusability of robot software parts by designing and implementing communication among a lot of software processes.

On the contrary, FPGA (Field Programmable Gate Array) is expected as an energy-efficient platform. However, it is difficult to introduce FPGA into a system in many situations since the productivity of circuit design is low. This is mainly due to poor portability of circuit modules (IPs: Intellectual Property) in current FPGA design environment.

We propose a method of designing a parallel distributed system on FPGA, based on Publish/Subscribe communication model. This paper discuss a method of realizing Publish/Subscribe communication on FPGA.

2. Framework for Publish/Subscribe Communication Model

2.1 Publish/Subscribe Communication Model

In Publish/Subscribe communication model, multiple processes communicate with each other. Processes communicate via logical channel called “Topic“. A subscriber (a receiver process) subscribes to a topic in advance, in which the subscriber is interested. When a publisher (a sender process) publish data to the topic, subscribers receive the data from the topic.

2.2 Requirements for the Framework

In order to realize a framework for Publish/Subscribe communication in circuit components on FPGA, it is necessary to satisfy the following minimum requirements.

(1) A process can receive data from a topic, to which the process subscribed in advance.
(2) A process can publish data to a topic.
(3) Multiple processes can publish or subscribe to a topic.

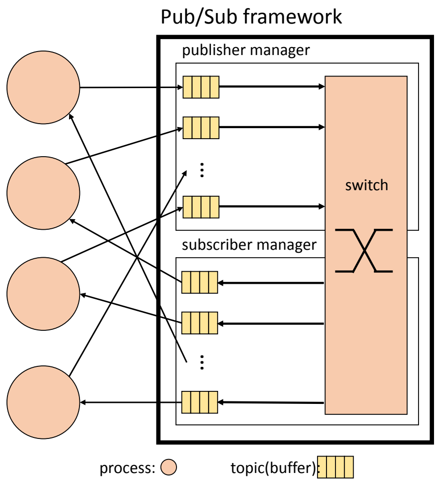

Fig. 1 Framework for Publish/Subscribe communication

Figure 1 shows the framework for communicating among processes. In our framework, a topic consists of a set of FIFO buffers that correspond to the publisher and subscriber, respectively. If the topic has M publishers and N subscribers, it employs M FIFO buffers for input, N FIFO buffers for output, and a switch function so that it offers $M \times N$ communication.

To communicate with this framework, publishers write data to upper FIFOs and subscribers read from lower FIFOs in the figure. The switch delivers data from upper FIFOs to lower FIFOs. This function satisfies the requirements of (1), (2). Also, by enabling $M \times N$ switching between FIFOs, $M \times N$ communication is realized. This function satisfies the requirements of (3).

3. Example using the Framework

3.1 Design of Parallel Processing System

As a design example using the above framework, we show a system that resizes an image. Image resizing is used in image recognition processing with local feature descriptor. Figure 2 shows the Publish/Subscribe communication model of the system. The system resizes an image to four resized images, which are different in size. A master process publishes an original image to a topic. Four worker processes receive the image and generate resized images at different scales respectively.

[1] Graduate School of Engineering Utsunomiya University, Yoto 7–1–2, Utsunomiya, Tochigi, 321–8585, Japan
[a)] kenta@virgo.is.utsunomiya-u.ac.jp
[b)] ohkawa@is.utsunomiya-u.ac.jp

3.2 System Architecture

Figure 3 shows the block diagram of the system that is designed with the proposed framework. This framework has 5 FIFO buffers for Publishers and 8 FIFO buffers for Subscribers. The number written on the FIFO buffers in the figure are the ids of topics. Topic 1 is an channel for an original image, whose publisher is the master and subscribers are workers 1 to 4. Topic 2 to 5 are channels for resized images, whose publisher is each worker and subscriber is the master.

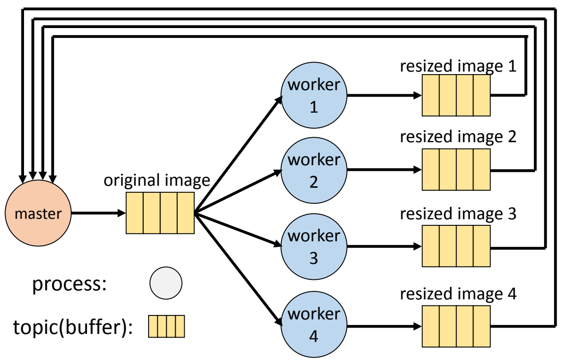

Fig. 2 Publish/Subscribe communication network of image resizing system

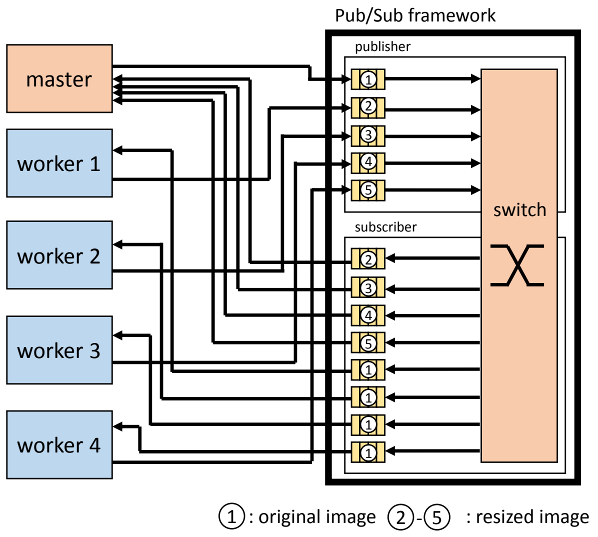

Fig. 3 Block diagram of image resizing system using proposed framework

4. Evaluation

4.1 Experimental Setup

This section describes the evaluation about the switch of the designed framework. The evaluation was made on hardware resource utilization and estimated clock period. In this evaluation, the proposed framework is implemented on Xilinx Kintex-7 FPGA (XC7K325T-2FFG900C), on Genesys 2 board [3] (Digilent inc.). The switch block was implemented with C++ language using Vivado HLS 2018.1 (Xilinx inc.).

4.2 HLS code example

Figure 4 shows a part of implementation. In this sample code, two publishers and three subscribers are connected as written at

```
ap_uint<8> buf;
void pubsub_framework(
    // topic for publisher
    hls::stream< ap_uint<8> > &ptopic0,
    hls::stream< ap_uint<8> > &ptopic1,
    // topic for subscriber
    hls::stream< ap_uint<8> > &stopic1,
    hls::stream< ap_uint<8> > &stopic00,
    hls::stream< ap_uint<8> > &stopic01,
){
#pragma HLS INTERFACE ap_ctrl_none port=return
    if(!ptopic0.empty()){ // topic 0
        buf = ptopic0.read();
        stopic00.write(buf);
        stopic01.write(buf); }
    if(!ptopic1.empty()) // topic 1
        stopic1.write(ptopic1.read());
}
```

Fig. 4 Source code of publish/subscribe communication framework

the arguments. Using type *hls_stream<>* in the function argument, Vivado HLS generates an interface for FIFO buffer. Pragma indicates that this function always execute without any control signal. The main body of the function checks if there is any data published in the publisher buffer. If data exists, the data is read from the publisher buffer and written to subscriber buffer.

4.3 Results

Table 1 shows hardware resource utilization. The data does not include resources of FIFO buffers. Both FF and LUT are less than 0.1% of the FPGA device.

Table 1 Hardware resource utilization

Name	FF	LUT
Expression	0	4
Multiplexer	-	130
Register	2	-
Total	2/407600	134/203800

The estimated clock period of the framework is 4.38ns. That is, the framework works at 200MHz which is the default frequency of the board.

5. Conclusion

This paper proposes a Publish/Subscribe communication framework for circuit components on FPGA. Instead of connecting circuit modules at electric signal level, we propose connecting processes of Publish/Subscribe model. To directly map Publish/-Subscribe model onto FPGA, we design a framework to realize Publish/Subscribe communication model of circuit components on FPGA. This framework is structured with FIFO buffers on the FPGA. Topics are assigned to FIFO buffers.

The future works are implementing the image resizing system and comparing the system with conventional implementation.

Acknowledgments This research was supported by the JSPS Kakenhi grant(15K00068, 16K00068, 17K00072).

References

[1] P. T. Eugster, P. A. Felber, R. Guerraoui, and A. M. Kermarrec, "The Many Faces of Publish/Subscribe", ACM Computing Surveys, vol.35, no.2, pp.114–131, June 2003.
[2] ROS official page, https://www.ros.org, (accessed July 5, 2018)
[3] Digilent Inc, "Genesys 2 Kintex-7 FPGA Development Board", https://store.digilentinc.com/genesys-2-kintex-7-fpga-development-board/, (accessed July 5, 2018)