A Cost-efficient FPGA-based Embedded System for Biosensor Platform

Iksu Jang, Jaeyoung Seo, Changjae Moon, and Byungsub Kim
Department of Electronics Engineering
Pohang University of Science and Technology (POSTECH)
Pohang, Korea
byungsub@postech.ac.kr

2022 19th International SoC Design Conference (ISOCC) | 978-1-6654-5971-6/22/$31.00 ©2022 IEEE | DOI: 10.1109/ISOCC56007.2022.10031540

Abstract— **This paper presents an improved cost-efficient FPGA-based embedded system for a biosensor platform. By sharing and incorporating the buffers in the data path, the proposed FPGA-based embedded system enables efficient and simultaneous data communication while reducing area and power consumption. We can simultaneously send bundles of 16-bit data owing to additional buffers in the data path without overwriting or waiting for previous data.**

Keywords; Biosensor platform; Embedded system; message passing communication

I. Introduction

For biomedical applications such as monitoring blood glucose levels for diabetics and a wearable sensor system, a cost-efficient biosensor system or platform has been developed. For example, a portable biosensor platform was proposed for ion-sensitive field-effect transistor (ISFET) and enzyme-based sensors [1], [2]. This biosensor platform utilized digital blocks which were implemented in the FPGA and also used digital-to-analog converters (DACs)/analog-to-digital converters (ADCs) to measure target concentration (Fig. 1).

However, the previously developed platform in [1], [2] is inefficient and consumes large power and area. The previously proposed biosensor platform [1], [2] used redundant data paths simultaneously to drive various or a bunch of sensors. The 24-bit output from ADC finite state machine (FSM) in the ADC controller was directly connected to USB interface without a data buffer such as the first input first output (FIFO). Because there is no such buffer between the ADC controller and USB interface, there is an unwanted timing interval to wait for finishing a previous task between an ADC's read command and receiving an ADC's output data. In addition, the number of buffers should be increased in proportional to the number of slaves in a conventional system, and thus the conventional system consumes more hardware cost and power.

In this paper, we propose an improved cost-efficient FPGA-based embedded system for a biosensor platform to drive various or massive sensors. The proposed system can reduce power and area consumption by sharing buffers in data paths efficiently. In addition, we can send data simultaneously without any chance of data overwriting because the proposed system has additional buffers at the output of the slave controllers.

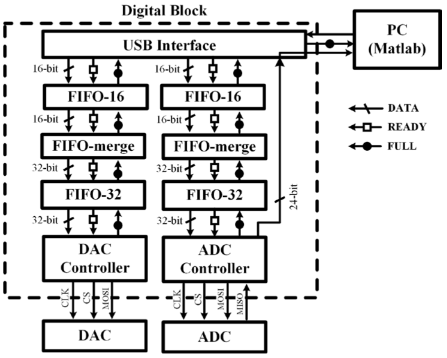


Figure 1. A block diagram of previously proposed digital block for biosensor platform in [2].

II. Proposed Digital Block

A. Overall Architecture

The proposed digital block which is implemented in the FPGA (OpalKelly XEM7010) consists of a USB interface, FIFO16, FIFO48, FIFO-merge, FIFO-split, MUX, DEMUX, and slave controllers (Fig. 2). The 48-bit message which is sent from PC (MATLAB) is split into three 16-bit packets. These three 16-bit packets are transmitted into the digital block through a USB interface. The three 16–bit messages are buffered in FIFO16 and merged into a 48-bit message by FIFO-merge. DEMUX receives the 48-bit message from FIFO48 and sends the 48-bit message to a slave controller according to the addresses of the 48-bit messages. The slave controllers are connected to output pins of FPGA to communicate with external modules such as DAC, ADC, LCD, or scan chain. Each slave controller is implemented differently depending on the slave module's communication interface such as serial peripheral interface (SPI) for DAC and ADC, 8-bit parallel communication for character LCD, or scan chain. Once the slave controller finishes the task according to the received 48-bit message, the slave controller sends a message back to MUX. Although no information is sent from the slave controller to the PC, the originally transmitted messages are always returned to the PC to verify whether the transmitted messages are successfully delivered or not and to check that the task is completed or not. MUX gathers messages from all slave controllers and transmits them to FIFO48. The buffered 48-bit message in FIFO48 is split into three 16-bit messages by FIFO-split. These three separate messages are queued in FIFO16. These FIFO48 and FIFO16

This work was supported in part by the Commercializations Promotion Agency for R&D Outcomes (COMPA) grant funded by the Korea government (MIST) (No. 2021I100); in part by the BK21 FOUR Project of NRF for the Department of Electrical Engineering, POSTECH; in part by the MSIT(Ministry of Science and ICT), Korea, under the ITRC(Information Technology Research Center) support program(IITP-2021-0-02052) supervised by the IITP(Institute for Information & Communications Technology Planning & Evaluation); and in part by Samsung Electronics Co., Ltd (IO201211-08055-01).

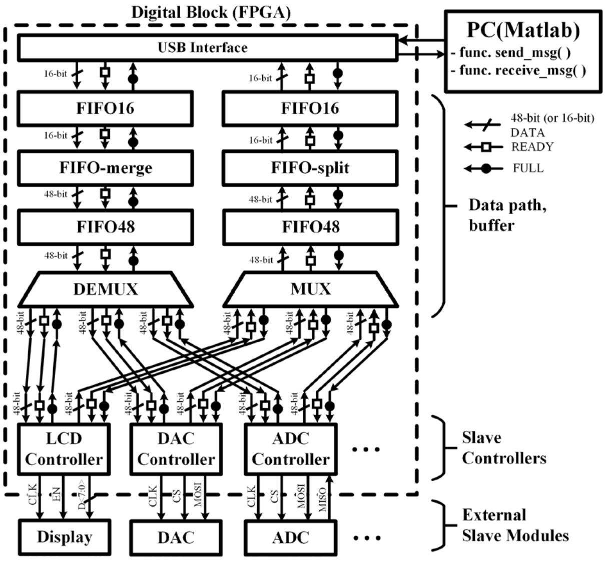


Figure 2. A proposed block diagram of a digital block.

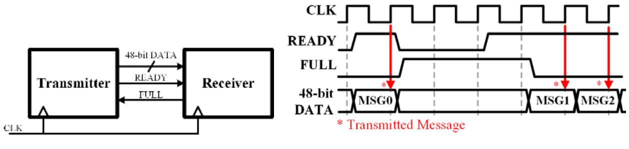
between MUX and USB interface can queue the consecutive output of slave controllers without worrying about data overwriting. Therefore, we can send multiple data at once without waiting for data to be received. By using a function in PC software (MATLAB), the number of 16-bit messages in FIFO16 is counted, and as many messages as the user wants are tranmitted to PC through the USB interface. Each received 16-bit message is converted into a 48-bit message. Finally, the user can check the received messages which are sent from slave controllers at the PC software. Thanks to DEMUX and MUX, all slave controllers can share the buffer in the data path: FIFO16/48, FIFO-merge, and FIFO-split. Therefore, the proposed system saves the area and power of digital blocks.

B. Message passing communication

In the digital block, 48-bit (or 16-bit) messages are transmitted to each other block using *1-bit READY*, and *1-bit FULL* control signals (Fig. 3). The *READY* signal becomes high when the transmitter is ready to send a message. The *FULL* signal becomes low when the receiver can receive a message. On the other hand, the *FULL* signal becomes high when the receiver block is working for other previously received messages or when its buffer is full. The message is only transmitted to other blocks when the *READY* is high and the *FULL* is low, simultaneously. If the transmitter's *READY* signal is high but the receiver's *FULL* signal is still high, the *READY* signal must be maintained high until the *FULL* becomes low. The examples of simulated waveforms are shown in Fig. 4.

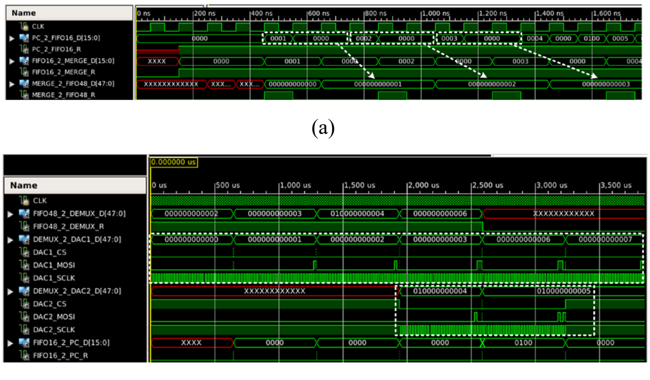
C. Composition of a message

The 48-bit message consists of a 32-bit Payload, 8-bit identifier (ID), 6-bit address, and 2-bit command (CMD). Payload is data to be transmitted to the slave controllers. ID is sequentially given to each transmitted message from MATLAB. Messages generated in MATLAB have their IDs. IDs are managed by an ID table that can recognize which messages were sent and which messages did not be sent back to the PC. Each slave controller processes the received messages in parallel, but the processing speed of each slave controller can be different. The unsorted received messages can be sorted in consecutive order by using ID. The address contains information whose message of a slave controller will be sent: each slave controller has a unique address. DEMUX transmits a message to the corresponding slave controller according to the address. CMD is reserved bits that can be used if necessary.



(a) (b)

Figure 3. Message passing communication in the digital block. (a) Block diagram with DATA, READY, and FULL signals. (b) A timing diagram of the data transfer in the digital block.



(a)

(b)

Figure 4. Simulated waveforms of a digital block with two slave controllers: (a) 16-bit data (PC_2_FIFO16_D) were transmitted and merged to 48-bit messages (MERGE_2_FIFO48_D) and (b) two slave controllers received messages (DEMUX_2_DAC1_D, DEMUX_2_DAC2_D) and generated SPI signals for external DACs. The transmitted messages were returned to PC (FIFO16_2_PC_D).

III. Conclusion

In this paper, we proposed an improved cost-efficient FPGA-based embedded system for the biosensor platform. The proposed system can communicate with a bunch of slave modules in both directions simultaneously. Due to the shared buffers in the data path, the digital block reduces area and power consumption.

Acknowledgment

The authors thank Jaehyun Ko and Changyoon Han for their valuable assistance.

References

[1] I. Lee, B. Kim, S. Lee, K. Lee, and H. Yi, "An FPGA-based embedded system for portable and cost-efficient bio-sensing: A low-cost controller for biomedical diagnosis," *IEEE International Circuits and Systems Symposium (ICSyS)*, Sept. 2015.

[2] I. Lee, S. Lee, K. Lee, C. Park, D. Kim, J. Lee, H. Yi, and B. Kim, "A reconfigurable and portable highly sensitive biosensor platform for ISFET and enzyme-based sensors," *IEEE Sensors J.*, vol. 16, no. 11, pp.4443-4451, Jun. 2016.