Proceedings of the 2023 International Conference on Mechatronics and Smart Systems
DOI: 10.54254/2755-2721/9/20230115

Theoretical analysis and comparison of memory copy accelerators

Shizhe Shen

Electronic Information Engineering Student Department, Shanghai University, Shanghai, 99 Shangda Road, China

shenbaby_2002@shu.edu.cn

Abstract. Over the last few years, more and more compute-intensive and memory-intensive applications have occurred. These applications are not able to be realized without the base of strong computing and high memory performance. The former has been developed successfully and outstrips the latter to certain degrees. So, researchers must overcome the shortcomings of memory copy performance to catch up with the standard of computing ability so that the systematic applications can be improved. This paper introduces two solutions for memory copy accelerators and analyses their advantages and disadvantages. It also presents an innovative floating window model to replace the channel tags model so as to boost efficiency. With these above-mentioned key components, a comprehensive system is developed, which is a cost-effective and error-resistant improvement of the conventional memory copy accelerators. The new improved memory copy accelerator can be a substitute for the present ones and give a foresee of the future of memory copy solutions.

Keywords: memory copy, DMA, Cache-Based, sliding window.

1. Introduction

In recent times, there has been a rapid increase in the usage of both compute-intensive and memory-intensive applications in recent years. Their value is discovered in various domains of medical informatics, autonomous driving, etc. These applications require not only powerful computing capabilities but also high-performance memory. Studies in previous years show that the developing speed of memory performance has not matched that of processor performance [1]. The memory process such as copying, comparing, moving, etc. needs to be performed by the host CPU, thus leading to performance degradation in many applications [2].

This paper presents two different kinds of memory copy accelerators (MCAs) to solve this problem. The first one is the Processor-DMA-Based Memory Copy Accelerator. Section 2 clarifies the operating principle, interconnection with the system, internal architecture and the rules followed by channel tags. Three novel ideas are proposed in this solution: a better location for the MCA, the multi-channel design and a more efficient management method using channel tags. The second solution is the Cache-Based Memory Copy Accelerator, which is elaborated on in section 3. It provides an innovative concept that transmitting the address where the data is stored instead of transmitting the whole data itself is practical, thus we can avoid cache pollution and save time which was used to copy the existing data in the cache to a new address. Section 4 involves two parts: the first is the proposition

DOI: 10.54254/2755-2721/9/20230115

of a comprehensive system that combines the advantages of the two solutions with some advice about how to aggregate in different situations. The second part elaborates on the sliding window model, conceived as improving the channel tag model. It noticeably decreases the resending latency thus improving the system cost and efficiency. This paper's conclusion will help researchers get a cost-effective and error-resistant solution for improving the conventional MCA.

2. Processor-DMA-Based Memory Copy Accelerator

2.1. Operating principle

Some basic theorems are stated in this section. First, the definition of DMA is an architecture that is an optional capability of an I/O controller. Once initiated by the processor, input/output (I/O) controllers that possess direct memory access (DMA) capabilities can directly access memory to retrieve commands and data and report their status [3]. It is a quite conventional concept that was proposed first in 1994. In other words, DMA is a feature offered by certain computer bus architectures that permits data to be transferred directly from an attached device, such as a hard disk drive, to the motherboard's memory. It was a rather successful and innovative way in a long period of speeding up computer operations because the microprocessor is freed from involvement with the data transfer [4]. The process is managed by a specific chip called a DMA controller (DMAC).

After knowing about the most traditional DMA method, an advanced DMA solution called Processor-DMA-Based Memory Copy Accelerator with many improvements aiming to address the disadvantages of the older DMA methods will be introduced.

2.2. Interconnection with the system

The system interconnection architecture is shown in Figure 1.

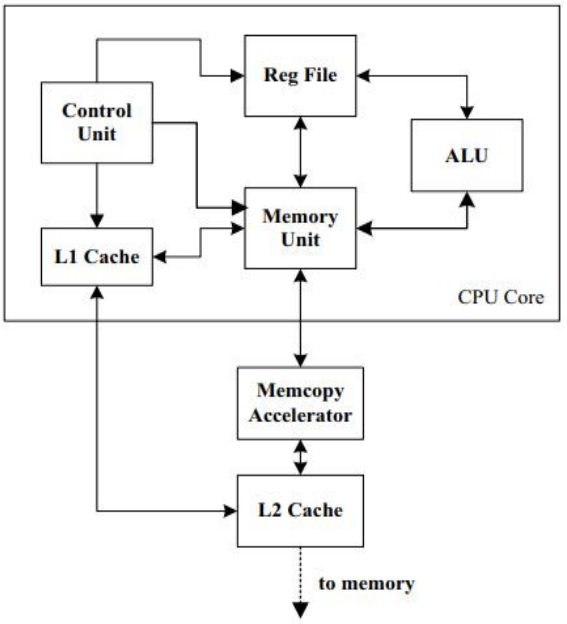

Figure 1. System interconnection architecture.

The Processor-DMA-Based Memory Copy Accelerator (Processor-DMA-Based MCA) is situated closest to the CPU core. It contains two AXI interfaces and a DMA engine.

Such a design has two advantages over the older ones.

Given that the distance between the two units that communicate with each other decides the cost of the communication, this design hugely decreases the cost by minimizing the distance. The Processor-DMA-Based MCA is placed outside the CPU core and the memory subsystem so it reduces the side

DOI: 10.54254/2755-2721/9/20230115

effects on the memory subsystem. Any problem happening in the subsystem will not impact the operation of the memory accelerator.

As Figure 1 shows, 2 caches called L1 and L2 are linked directly. They will be responsible for storing the data that DMA will read or write with direct access. As a result, data no longer need to go through the memory unit in the CPU core and the memory accelerator outside the CPU core. It can undoubtedly decrease the average latency.

2.3. Internal architecture

According to some descriptions of the conventional DMA, there are two main disadvantages [5].

(1) High setup overhead. Before each data transfer task, they need to set up a DMA channel control logic (CCL), which requires communication at least 3 times.

(2) Disability of addressing abnormal circumstances. Because the data transfer progress is atomic and invisible to software in the conventional DMA, it is hard for the users to precisely locate the error through software.

Luckily, the Processor-DMA-Based MCA solves these 2 problems as well. Figure 2 shows the internal architecture of processor DMA.

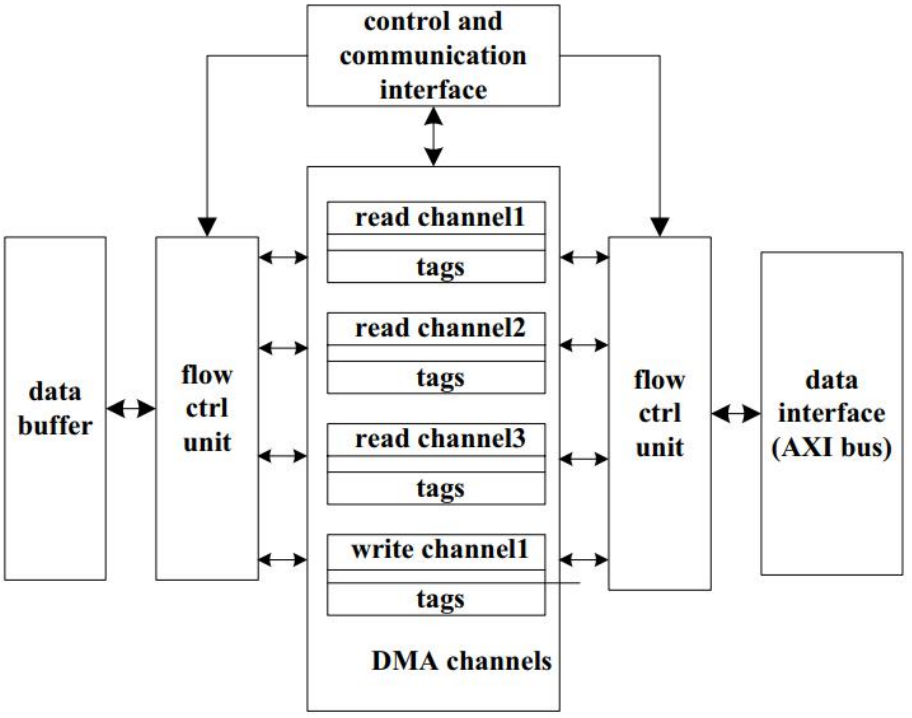

Figure 2. The internal architecture of processor DMA.

It can support:

(1) High-speed data transfer capacity & memory read and write operations that are pipelined.

(2) Light overhead process for channel switching, setup, and data transfer stalls or restarts.

(3) A channel-specific reconfigurable interface for the control and communication during DMA data transfer.

The specialty of the architecture is that it increases the DMA channels which can work concurrently. There are 3 read channels and 1 write channel. This method can noticeably diminish the time when channels start, switch, or reset since the trickiest setup progress can be used for all channels.

The second problem is the Disability of addressing abnormal circumstances. The architecture uses DMA channel tags to solve it. The tag is used to identify the progress of data transfer. Detailed information on the exceptions will be involved in the tag received in the flow control unit. Session 2.4 clarifies the concrete method of how we use channel tags.

DOI: 10.54254/2755-2721/9/20230115

2.4. Rules followed by channel tags

In this session, the rules DMA channel tags follow and how they can convey error messages to users are elaborated. A DMA tag contains 2 parts. An s-tag and a p-tag. They are all 8 bits. S-tag performs the channel control and p-tag involves the information about the data transfer trace. The data transfer progress is divided into 8 parts (part 0 to part 7), which have a mapping relationship with the 8 bits in the tag from the LSB (Least Significant Bit) to the MSB (Most Significant Bit). S-tag represents the state of the transmission interface, and p-tag represents the state of the reception interface.

For example, if the transmission interface has sent 1/8 of the entire data, the value of the s-tag should be 0000_0001 at that time, and the value of the p-tag is still 0000_0000. Only when the receiving port receives the data and detects that it is correct, the p-tag will become 0000_0001, the same value as the s-tag. In this way, the value of the s-tag will change from 0000_0000 to 1111_1111, and the p-tag will do the same but have a latency. Figure 3 below shows the concrete time latency between the p-tag and the s-tag.

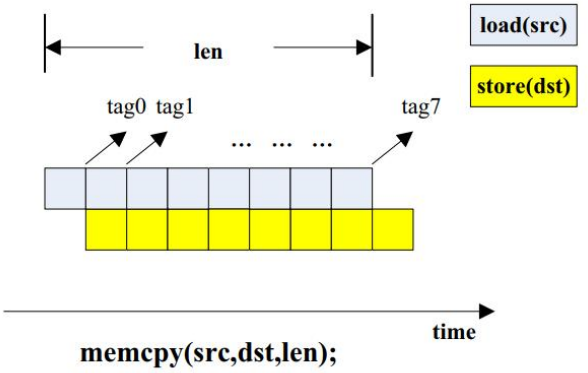

Figure 3. Pipelined memory copy progress.

Strictly obeying these rules, the memory accelerator works in the following sequences:

(1) The CPU sets up the s-tag and DMA channels for each channel.

(2) Data flow across DMA channels begins.

(3) During each cycle, DMA initializes the p-tag and verifies the s-tag.

If it meets the requirement specified by the CPU in step (1), DMA transmits a signal to the CPU and proceeds to step (4); otherwise, it goes back to step (2).

(4) The CPU verifies and then configures the p-tag and proceeds to the next stage of data transfer, or terminates the transfer altogether.

3. Cache-Based Memory Copy Accelerator

Another great solution to speeding up the memory copy progress is expounded in this section. Although the Processor-DMA-Based MCA has raised two solutions to two major disadvantages of the conventional DMA, it still has some limitations.

Firstly, DMA controllers (DMAC) are peripheral devices, although Processor-DMA-Based Memory Copy Accelerator spares no effort in minimizing the distance between the CPU core and DMAC, there is still an unavoidable cost of the communication.

Secondly, although the number of channels is increased from 1 to 4 to make full use of the initialization, i.e., the 4 channels can work simultaneously after one initialization, there is still a significant overhead on the communication, as it needs to perform the initialization explicitly.

Third, the Processor-DMA-Based Memory Copy Accelerator uses several tags in the 4 channels to transfer the message of competition. During this process, an interrupted way is used, and we know that either the polling way or an interrupted way is expensive [6].

DOI: 10.54254/2755-2721/9/20230115

To find a solution that can effectively perform memory copies by exploiting the presence of a cache, some scholars have done much research. It can be observed that most of the data requiring memory copying is already cached, thus we can use these data directly instead of moving them out from the cache to a series of new addresses [7]. A Cache-Based Memory Copy Accelerator (Cache-Based MCA) is an example of this concept.

3.1. Operating principle

Initially, the paper introduces the notion of an indexing table linked to a direct-mapped cache. Readers should acknowledge an assumption that the original data has been already stored in the cache. It is a great thought that we can utilize an indexing table with the copied data address as its index to memorize the address containing the original data. With such a concept, the conventional memory copy method can be replaced with a new solution that inputs the address of the copied data into the indexing table and establishes a pointer to the initial information that is cached.

3.2. Internal architecture

Figure 4 shows the internal architecture according to the accelerator concept [8].

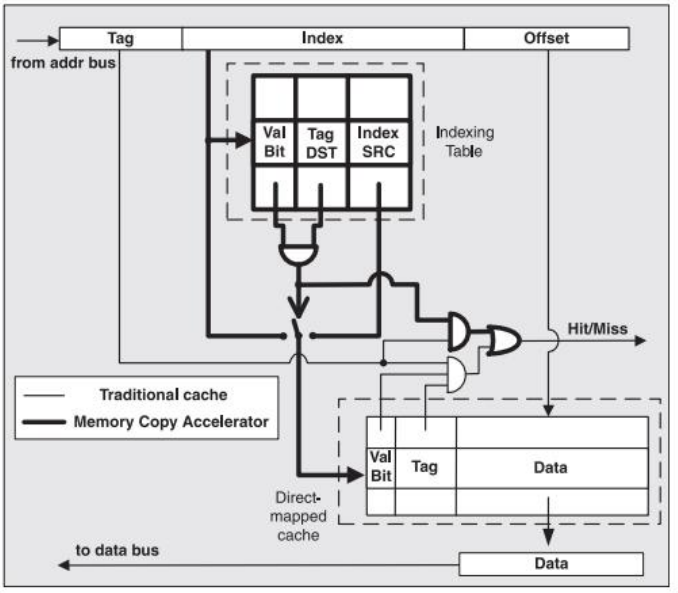

Figure 4. The internal architecture of Cache-Based MCA.

The main part of the Cache-Based MCA is the direct-mapped cache, which is presented with the thinner lines, thinner gates, and the table above in Figure 4. It is divided into 2 parts: a cache directory and cache data memories. The main address of the data included in the appropriate place of the cache data memory is contained in the cache directory. From the address bus, we will receive a piece of address information which is made up of 3 parts: tag, index, and offset.

They have different functions clarified as follows [9].

(1) Tag. The tag has 2 functions varying from the writing process to the reading process. In the writing process, it is written into the cache directory and in the reading process, it is used to make a comparison with the existing tag in the cache directory.

(2) Index. Locating the cache directory and cache data memory will be done using the index.

(3) Offset. The offset provides what the word is finally to be supplied to the processor.

During the process of using the index to locate data, there are two different circumstances. Whether the valid bit is set or not decides.

(1) The valid bit in the corresponding entry of the table is set. In this circumstance, 2 sub-circumstances are remaining and we should use the tag part to determine which one is suitable. (i) The

DOI: 10.54254/2755-2721/9/20230115

tag stored on the table matches the tag portion of the address given by the processor. The result of the operation will provide the index segment of the source address, which is utilized to access the cache. (ii) The tag recorded on the table does not match the tag portion of the address given by the processor.

(2) The valid bit in the corresponding entry of the table is not set. In this scenario, the index segment of the processor's address is utilized to access the cache.

4. Integration and improvements

4.1. Integration

After knowing about the two solutions of memory copy accelerators, it occurs to me that a complete system can use both of them to get better performance. My suggestions are as follows:

Keeping the memory copy process for the transmission of vital data, because it is a more conventional as well as safer method, which can avoid all the incidents such as the error of the address or the accidental power outage. In this way, we need an MCA. The minimization of the distance between the memory accelerator and the CPU core and multi-channel design which can help us utilize the initialization many times will help.

For the other relatively less important data, the Cache-Based MCA seems a better choice. It can help avoid the cache pollution problem created by the conventional memory copy process. Although Cache-Based MCA has the name “memory copy”, it turns the process of transmitting the complete data into the process of transmitting the address of the data, so the data itself is avoided being copied and pasted a lot of times [10]. What we need is just the address of the data in the situation where we need the complete data, so the file size needed to be transmitted is hugely decreased.

4.2. Improvement

Last but not least, an improvement of the Processor-DMA-Based MCA is shared. As Figure 3 shows, we use an 8-bit tag in the pipelined process, the tag is used to present the successfully done portion of the transmission job and the reception job. After an 8-bit tag switching from 0000_0000 to 1111_1111, we know that a message has been successfully sent, but when we meet that the value of the s-tag is not equal to the p-tag, which means some errors happened during the transmission process and the reception port has got a wrong data, the current solution chooses to resend all the data, at the same time reset the s-tag and the p-tag back to 0000_0000, then make the transmission again for the message. This is a high-cost solution because maybe only 1/8 of the message goes wrong, but we need to resend the whole message.

The Sliding Window Protocol provides me with a new idea. After plenty of research and experiments, this paper integrates the sliding window theory with the existing tag solution. A sliding window protocol is a feature of packet-based data transmission protocol. It can minimize the data resent to the least, only the data has errors. In the 8-bit example, that is 1/8 of the complete data. Accordingly, it is used to improve the efficiency of a channel with high latency.

Figure 5 shows the construction of the window.

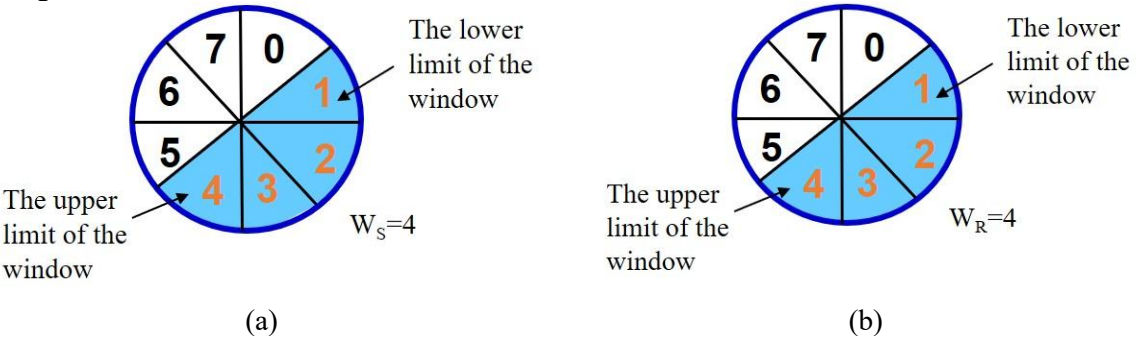

Figure 5. window construction: (a) sending window (b) receiving window.

DOI: 10.54254/2755-2721/9/20230115

A detailed example of how it works is presented in Figure 6. The time order of the occurrence of each status is from t0 to t7. The V(S) and V(R) show the lower limit of the window.

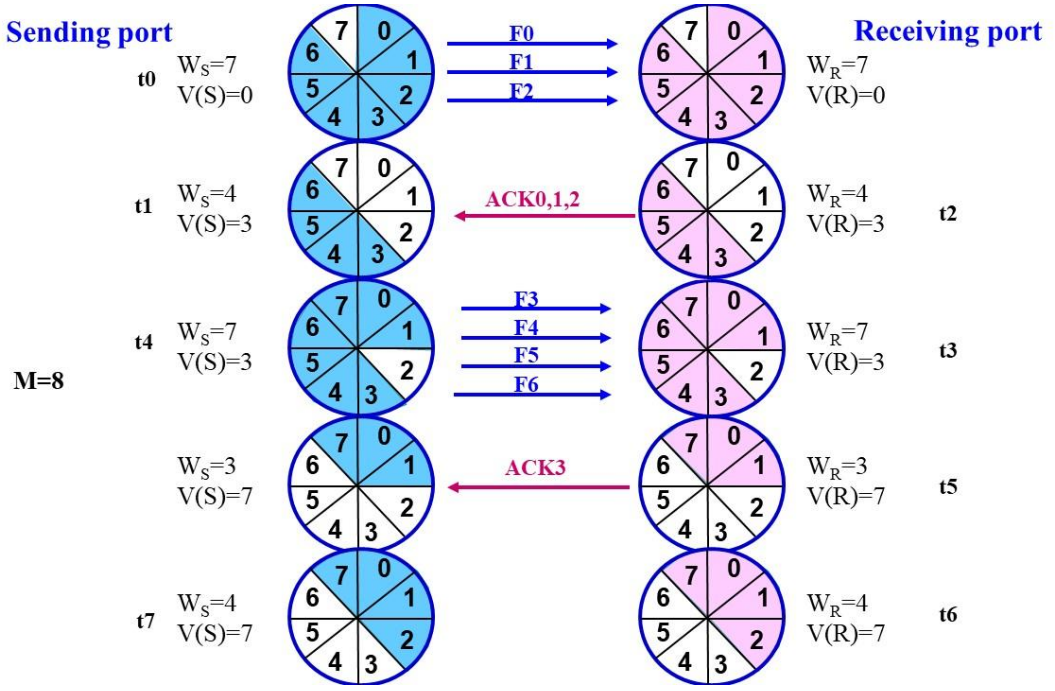

Figure 6. An M=8 data transmission work in the sliding window model.

5. Conclusion

The paper clarified two different kinds of memory copy accelerators. The first one is the Processor-DMA-Based Memory Copy Accelerator, which decreases the cost of communication, improves the concurrent transmission efficiency and makes high use of the initialization. The second one is the Cache-Based Memory Copy Accelerator, which solves the cache pollution problem and substantially improves the efficiency of transferring data. In a comprehensive system, two solutions can be used cooperatively. Users can freely choose the proper solution for different data according to their importance. And with the improvement proposed in section 4, readers can solve the transmission error by only resending 1/8 of the whole message, it can help the system diminish the huge number of messages resent and the total time of the transmission process thus drastically improving the system efficiency.

In the future, the memory copy accelerators will develop in the direction of increasing the copying accuracy and reducing the time and the CPU cores in demand. Solutions without the copying of the memories are a better choice, for example, sharing them directly with all the hardware devices.

References

[1] S. Wong, F. Duarte and S. Vassiliadis, "A hardware cache memcpy accelerator," 2006 IEEE International Conference on Field Programmable Technology, Bangkok, Thailand, 2006, pp. 141-148.

[2] T. Kalkhof and A. Koch, "Direct Device-to-Device Physical Page Migrations in Multi-FPGA Shared Virtual Memory Systems," 2022 32nd International Conference on Field-Programmable Logic and Applications (FPL), Belfast, United Kingdom, 2022, pp. 225-234.

[3] "ISO/IEC/IEEE International Standard for Information technology--Microprocessor systems--Control and Status Registers (CSR) Architecture for microcomputer buses," in ISO/IEC 13213: 1994, vol., no., pp.1-144, 5 Oct. 1994.

DOI: 10.54254/2755-2721/9/20230115

[4] B. Zeng, “What is Direct Memory Access (DMA)? - Definition from WhatIs.com,” WhatIs.com, 2022. https://www.techtarget.com/whatis/definition/Direct-Memory-Access-DMA

[5] W. Su, L. Wang, M. Su and S. Liu, "A Processor-DMA-Based Memory Copy Hardware Accelerator," 2011 IEEE Sixth International Conference on Networking, Architecture, and Storage, Dalian, China, 2011, pp. 225-229.

[6] F. Duarte and S. Wong, "Cache-Based Memory Copy Hardware Accelerator for Multicore Systems," in IEEE Transactions on Computers, vol. 59, no. 11, pp. 1494-1507, Nov. 2010.

[7] S. Vassiliadis, F. Duarte and S. Wong, "A Load/Store Unit for a Memcpy Hardware Accelerator," 2007 International Conference on Field Programmable Logic and Applications, Amsterdam, Netherlands, 2007, pp. 537-541.

[8] U. S. Solangi, M. Ibtesam, M. A. Ansari, J. Kim, and S. Park, “Test Architecture for Systolic Array of Edge-Based AI Accelerator,” IEEE Access, vol. 9, pp. 96700–96710, 2021.

[9] F. Duarte and S. Wong, "A memcpy Hardware Accelerator Solution for Non Cache-line Aligned Copies," 2007 IEEE International Conf. on Application-specific Systems, Architectures and Processors (ASAP), Montreal, QC, Canada, 2007, pp. 397-402.

[10] W. Muła and D. Lemire, “Base64 encoding and decoding at almost the speed of a memory copy,” Software: Practice and Experience, vol. 50, no. 2, pp. 89–97, Nov. 2019.