# SMEFF: A Scalable Memory Extension Fabric for FPGA

Wei Li, Yangyang Zhao, Yuhang Liu, Mingyu Chen
State Key Laboratory of Computer Architecture, Insitute of Computing Technology, Chinese Academy of Sciences
University of Chinese Academy of Sciences,
Beijing, China
liwei2012@ict.ac.cn, zhaoyangyang@ict.ac.cn, liuyuhang@ict.ac.cn, cmy@ict.ac.cn

***Abstract*—In resource-constrained FPGA systems, off-chip memory plays an important role in both prototype verification and acceleration systems for big data. As the scale of applications become increasingly large and complex, the data to be processed grows exponentially. In contrast, FPGAs provide limited memory capacity and bandwidth, severely limiting the scale and performance of prototype verification systems and acceleration systems. Furthermore, data movement is expected to be a dominant consumer of energy, thus inefficient data movement between different DRAM modules also incurs significant performance and energy penalties.**

**This paper proposes a practical design: A Scalable Memory Extension Fabric for FPGA (SMEFF), which is an asynchronous memory access mechanism and exploits cascaded technology to solve the problem of memory capacity and bandwidth. SMEFF uses two key technologies to achieve memory capacity and bandwidth improvements, and shrink the latency and overhead of data movement--the first is an FPGA-based high-speed serial bus to build a multi-level memory fabric instead of the traditional parallel bus mechanism to solve the signal integrity problem. The second is a module to module (M-To-M) DMA data movement technology, which reduces the latency and overhead of data movement between memory modules. We implement SMEFF on an FPGA-based prototype to demonstrate the feasibility of our approach. Experimental results show that SMEFF provides 5x memory capacity increase and up to 3.6x memory bandwidth improvement compared to state-of-the-art FPGA-based memory systems, and outperforms PCIe-based systems. The data movement of M-TO-M's DMA technology obtains up to 3x latency reduction, and average of 21.1% to 61.1% energy reduction compared to state-of-the-art FPGA-based memory systems. SMEFF thus increases FPGA-based prototype memory systems capacity and bandwidth. More importantly, our architecture provides opportunities for the design of scalable, cost-effective FPGA-based memory subsystems.**

## I. INTRODUCTION

As the scale of applications become larger and more complex, the data to be processed is growing dramatically, especially in big data systems and cloud computing systems. Meanwhile, using FPGAs as prototype verification systems or acceleration systems has become mainstream. Accordingly, FPGA-based prototype verification projects and acceleration systems require tremendous memories to get data with acceptable latency. Furthermore, state-of-the-art FPGAs only support 4 memory channels [1], which limits overall capacity. Especially, memory systems of FPGA-based solutions face particular difficulties in terms of capacity and bandwidth to meet these requirements [2].

To solve the traditional CPU memory wall [3] [4] [5], buffer chips can be used to mitigate pin count and SI limitations [6] [7] [8], and state-of-the-art research uses packet-based asynchronous protocols [9] [10] [11] [12] [13] [14] to extend memory. We believe that an industry-standard asynchronous interface is the right way to proceed [15]. It is comparatively easy to use FPGAs to implement asynchronous memory as compared to traditional CPUs. However, there is little effort focused on FPGA-based memory extensions, which is the main method for prototype verification systems, especially in cloud computing and big data domains. There are very limited ways to extend memory for FPGAs and state-of-the-art FPGAs only support 4 channels and face difficulties in extending on that scale [1]. The speed of the PCIe-based memory extension systems [13] are limited by the PCIe protocol, incurring large penalties in terms of delay and block-level transfers. At present, HMC [16] and HBM2 [17] are expensive, with limited capacity. In the short term, this makes them unsuitable for large-scale memory extension. Intel's SMB [18] and Cisco's UCS [19] are aimed at generic CPUs, and are not open standards. This makes them also unsuitable for FPGA systems.

We propose a practical solution to capacity and bandwidth that is much more scalable and requires no changes to the conventional FPGA-based memory system. It only utilizes an FPGA-based high-speed serial bus to provide higher capacity and bandwidth. In terms of cost, SMEFF uses low-cost FPGAs as buffer chips. Each buffer chip connects multiple memory channels (using low-cost UDIMM), and utilizes a high-speed serial bus to connect all nodes to form a scalable fabric. SMEFF uses packets to transmit data between memory controllers and buffer chips, and large packets can improve the utilization of data channels and reduce latency.

We summarize the key contributions of this paper as follows:

1) We propose a scalable memory extension fabric: a Scalable Memory Extension Fabric for FPGA (SMEFF), introduces the concept of a multi-level and scalable

978-1-5386-2656-6/17/$31.00 © 2017 IEEE

asynchronous memory fabric connected via high-speed serial bus.

2) We implemented the design of SMEFF: we exploit an FPGA-based high-speed serial bus to build a memory network. SMEFF takes advantage of low-cost FPGAs as buffer chips. Each buffer chip uses a high-speed serial bus to connect all nodes to form a scalable fabric. SMEFF provides 5x memory capacity increase and up to 3.6x memory bandwidth improvement compared to state-of-the-art FPGA-based memory systems.

3)The data movement of M-TO-M's DMA mechanism: we propose the module to module (M-TO-M) DMA near-data copy mechanism which offers 3x performance improvement and 21.1% to 61.1% energy reduction compared to state-of-the-art FPGA-based memory systems. It also outperforms PCIe-based extended memory systems.

We present some background on memory extension and discuss several kinds of state-of-the-art solutions in Section II. We propose Scalable Extension Fabric for FPGA-based system (SMEFF) in Section III. We explain our evaluation methodology in Section IV, and present results in Section V. We highlight some related work and conclude in Section VI.

## II. Background and Related Work

DRAM (Dynamic Random Access Memory) has the best tradeoff between speed, density, and cost as compared to other storage media such as SRAM, NVM and Flash [20] [21] [22] [23]. DRAM memory systems usually consist of multiple channels. Total memory capacity and bandwidth is thus determined by the number of channels.

The number of channels in the memory controller is limited by processor or FPGA's pin count. A standard DRAM channel contains more than 100 signal lines [24]. As a result, current processors and FPGAs typically support 2 to 4 memory channels. However, due to chip packaging technology, the number of pins only increases by about 6.5% every year [25], thus limiting the number of channels, and memory capacity and bandwidth. Hybrid Memory Cube (HMC) [16] [26] [27] and High Bandwidth Memory (HBM) [17] are still capacity-limited and expensive. Currently, the two main approaches of memory extension are synchronous access and asynchronous access.

### *A. Synchronous access*

In terms of memory capacity and bandwidth limitations, signal integrity and pin count are important. The signal integrity problem can be solved by adding buffer chips. In reality, LRDIMMs are widely adopted memory solution for large-scale applications [25]. LRDIMM uses a buffer chip to increase the drive capability and reduce the impact of signal integrity issues. However, the capacity is still limited. Moreover, large capacity LRDIMMs are at least 4 times more expensive as compared to UDIMM of the same capacity. Other emerging technologies are still not available because of immaturity and cost.

Intel's SMB [18] (Scalable Memory Buffer) memory system uses the ASIC buffer to extend DDR3 DRAM channels. SMB extends a custom serial bus to two standard DRAM buses. However, such buffer chips with more DDRx interfaces become expensive due to large die area and packaging costs. Intel is unlikely to open relevant technology, so SMB is difficult to port to other large-scale applications. With a similar concept, the Cisco's UCS [19] system uses buffer chip to extend one standard DRAM bus to four separate buses, which can obtain 4 times capacity. However, due to large footprint of the buffer chip and high cost of production, this technology is not public, so it is difficult to apply to FPGA-based memory extension.

FBDIMM also uses synchronous protocols to expand memory capacity. The capacity of memory systems can be extended through a dedicated memory controller and multi-cascade buffers [20] [28]. Dedicated memory controllers can tolerate different delays in the multi-level FBDIMM cascades, and theoretically has better scalability. However, current processor manufacturers no longer support FBDIMM, thus limiting the scope of application factors, since it has become very difficult to buy and expensive.

HBM2 [17] uses vertical stacked nodes and high transmission bandwidth, with innovative small size and high performance graphics card applications. FPGA vendors have already supported HBM applications. However, HBM2 is limited in capacity and expensive, thus making it unwise to choose HBM as a memory extension in the immediate future.

### *B. Asynchronous access*

Technically, asynchronous interface protocols are an ideal solution for avoiding memory expansion barriers and can even inherit new storage media with greater advantages in density and power consumption. Cooper et al. [7] proposed the BOB (Buffer-on-Board) memory systems. In BOB, a buffer chip is placed between the processor and memory. The processor uses a serial bus to communicate with the buffer chip which is based on asynchronous access protocols and a synchronous communication protocol between the buffer chip and memory. Unfortunately, this method has limited extension capabilities in practice. This method is only proposed in theory, and has not been realized, thus making it unsuitable for FPGA prototype verification systems.

PCIe is a widely-used, packet-based, open standard asynchronous protocol. Lim et al [11] [12] exploits the PCIe interface as a memory extension system. DMA operations quickly exchange pages to and from the memory through the PCI interface. The speed of the PCIe-based memory extension system [13] is limited by PCIE protocol, incurs large delay penalties and only supports block transfers, not a standard 64B memory interface.

HMC [15] [16] uses 3D stack technology, with layers of both DRAM die and logic chips encapsulated in a single chip. The processor accesses the logical die using a customized asynchronous protocol. HMC has high bandwidth. Currently, HMC is expensive and limited in capacity. NVM is also too nascent to use as a memory replacement [29] [30] [31].

Chen et al. [32] proposed MIMS (message-based storage system). MIMS implements asynchronous access by adding

buffered chips. It utilizes semantic information to access data, can send longer messages, and achieve high data transfer efficiency. Still, it remains a research proposal.

### C. Conclusion

In this work, we extend memory in a similar cascaded fashion. We propose a scalable memory fabric (SMEFF). SMEFF uses an FPGA-based high-speed serial bus to build a memory network to solve memory capacity and bandwidth problems. The M-TO-M DMA near-data copy mechanism can reduce memory latency and power consumption issues.

## III. SMEFF Design

This section describes the designs of the SMEFF hardware and logic, which provides some details about: fabric design, M-controller design, MEB design and M-TO-M DMA mechanism.

On aspect of feasibility, the FPGA provides some high-speed serial buses. For example, Xilinx's new Virtex UltraScale+ FPGA XCVU37P [33] provides 96 pairs of high-speed serial connections. The transmission speed of each serial bus is up to GTY 32.75Gb/s [34] and total bandwidth is up to 96*32.75Gb/s = 3144 Gb/s. It is fully feasible to meet the requirements of most DDR3 and DDR4 transmission speed and capacity extension.

### A. Fabric Design

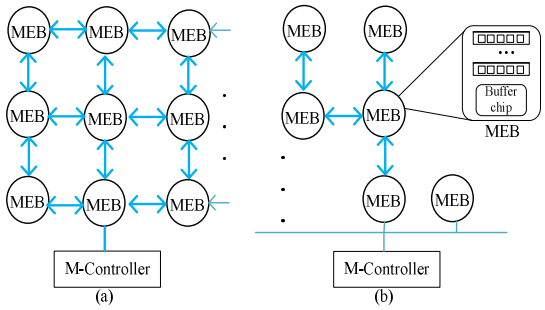


Fig. 1: Fabric of (a) mesh and (b) tree.

Fig. 1 shows the mesh topology and tree topology of the memory extension fabric. This structure uses memory extension boards (MEB) as nodes, with multiple MEBs and FPGAs interconnected to form a network. Utilizing an asynchronous access protocol, all nodes (MEBs) are interconnected via a high-speed serial bus. The SMEFF design is divided into two parts: the M-controller and the MEBs. For the M-controller implementation, we designed a scalable memory controller that can connect multiple MEB via a high-speed serial bus. For the MEB implementation, we use low-cost FPGA as a buffer chip. Each buffer chip as a node connects multiple memory channels. All nodes are connected with each other through high-speed serial buses to form a memory extension fabric.

### B. M-controller Design

In order to obtain high memory capacity and bandwidth, the M-controller consists of three modules as follows: memory extension controller (SMEFFC), SMEFF dispatcher and local dispatcher. This is shown in Fig.2.

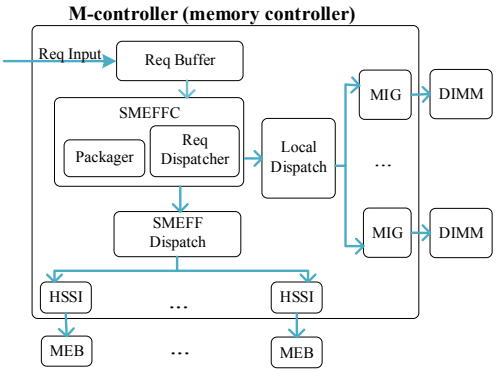


Fig. 2 The design of M-controller

The memory extension controller (SMEFFC) receives read requests and write requests from the request (Req)-Buffer. The Packager module of the SMEFFC then packages multiple requests into different packets based on the range of the mapped addresses. The dispatcher then distributed packets to the local dispatcher or the SMEFF dispatcher according to the destination address.

For local memory requests, we designed the Local-dispatcher. The local-dispatcher receives packets sent by the SMEFFC, and parses packets into read and write requests. The local-dispatcher distributes read and write requests in parallel to different local MIGs according to address ranges. Here, we do optimizations to merge memory requests from the same row, taking full advantage of the OPEN-BANK feature and reducing the number of reopened banks.

For extended memory requests, we designed the SMEFF-dispatcher. The SMEFF-dispatcher receives the packets sent by the SMEFFC and then parses packets into read and write requests. SMEFF-dispatcher distributes read and write requests in parallel to the different high-speed bus ports according to the address range. Then, the Routers of MEB forward them to their destinations. Here, we implement two destination addresses: the destination MEB address and the destination channel address. According to the destination MEB address, the packet is first routed to the corresponding MEB, and then according to the destination channel address, we find the corresponding channel.

When there is poor locality of data, the requests may be packaged into different packets. To improve the parallelism of memory access, Local-dispatcher or SMEFF-dispatcher distributes read and write requests to different memory controllers, and executes them in parallel. When there is good locality of data, a large chunk of data can be processed at a time, thus reducing access latency and enhancing bandwidth utilization. SMEFFC receives data returned from all memory channels, and uploads it to the Req-Buffer module.

### C. MEB design

The memory extension board (MEB) has two functions: first, if the destination MEB address of a packet is a local MEB request, MEB receives the packet and parse it into read

and write requests. According to the address range, MEB distributes read and write requests to different memory controllers of local MEB. Second, if the destination MEB address of a packet is not a local MEB request, the packets will be forwarded to the destination MEB by looking up the routing table. This is shown in Fig. 3 and Tab. 1.

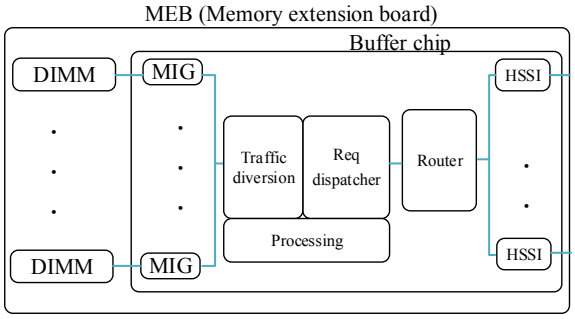


Fig. 3 The design of memory extension board

Tab. 3 shows the memory extension board (MEB) logical structure, which contains the following modules: Router, Request dispatcher, Traffic diversion, High Speed Serial Interface (HSSI) and MIG.

The primary function of the router is to select the direction of packet forwarding according to the destination MEB address (or ID). If the destination MEB ID is the local MEB, the packet is sent for local logical processing. Otherwise, the packet is forwarded to the destination port according to the routing table.

TABLE I. THE DESIGN OF ROUTING TABLE

| **MEB** address (or **ID**) | **Port number** |
|---|---|
| 1 | Local port |
| 2 | 3 port |
| 3 | 5 port |
| … | … |

As shown in Tab. 1, the routing table is divided into two columns, one for the destination MEB ID, and the other for the outgoing port corresponding to the different destination MEB.

For local MEB requests processing, we designed the Request dispatcher. The main function of the Request dispatcher is to receive the packets sent by the Router. The read request or write request is parsed from the packets, and then the requests are sent to the corresponding local memory controller according to the address range of the request. Simultaneously, the local memory controller returns the request data, which is packaged and sent back to the Router module.

The advantage of asynchronous protocols is the ability to process multiple read requests and write requests concurrently to maximize the utilization of the high-speed serial buses. The processing module can merge memory requests from the same row and reduce the number of open banks. The processing module also has the request scheduling function, which can configure different priorities according to different strategies, such as small request priority, FCFS, and read request priority.

The High Speed Serial Interface (HSSI) module provides extended memory with a set of High speed serial buses, which operates in full duplex mode. The Aurora IP core provided by Xilinx is used to send and receive data.

### *D. M-to-M DMA Design*

Currently, data movement is expected to be a dominant consumer of energy, and inefficient data movement also produces significant performance and energy penalties between different DRAM modules. When we perform a large copy, such as moving 1GB of data from memory module 1 to memory modules 2, 3, the current FPGA memory controller (such as Xilinx or Altera's memory controller) performs a page-by-page copy of data from memory module 1 to the memory controller which is then forwarded to memory module 2. After copying 1GB of data from memory module 1 to memory module 2, the same process goes from memory module 1 to memory module 3. The M-TO-M DMA mechanism can move data directly between MEB1, MEB2 and MEB3 in parallel, without consuming bandwidth of FPGA memory controller. This is shown in Fig. 4.

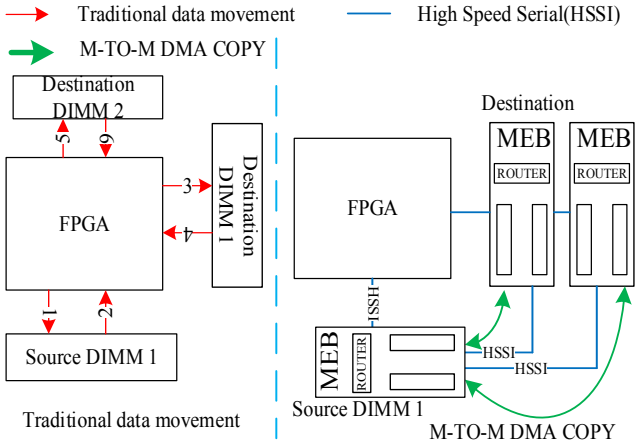


Fig. 4 Comparison of traditional memory operations with M-TO-M near-data copy

To minimize the number of opened banks during M-to-M DMA mechanism, the buffer size should be selected carefully. We assume the size of data to be copied is M, the data buffer block is m and the size of one physical memory page is p.

As shown in Figure 4. Function 1 indicates the number of pages to be opened during M-to-M DMA mechanism, if m is larger than p. Function 2 indicates the number when m is smaller than p.

$$2\times[M/p] \qquad \text{Function 1}$$

$$2\times[p/m]\times[M/p] \qquad \text{Function 2}$$

Symbol [] means round up to the upper integer. So if m is smaller than p, a page will be opened [p/m] times during the DMA process.

Based on the above analysis, the data buffer for the copy module is set to a page size of the physical memory.

## IV. Implementation of SMEFF

We implemented SMEFF. For the M-controller we used a Xilinx Virtex7 XC7VX690T [35]. We make motherboard to achieve SMEFF hardware. For the implementation of MEB, SMEFF uses low-cost FPGA (Virtex6 XC6VLX240T) [36] as a buffer chip. We make memory extension board (MEB) to achieve SMEFF memory expansion.

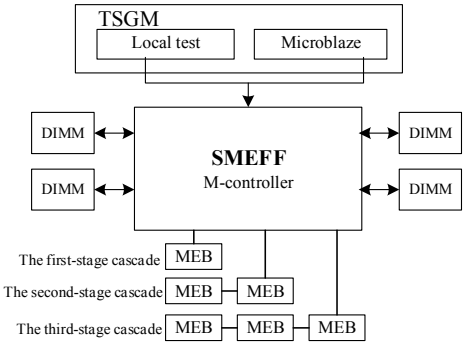


Fig. 5 The implementation of SMEFF

The implementation of SMEFF provides connections to 4 local memory channels, and 6 memory extension boards (MEB). Each MEB connects to 3 memory channels, so the entire design connects to 22 memory channels. This is shown in Fig. 5.

### A. Motherboard for the implementation of M-controller

In order to verify the feasibility and effectiveness of SMEFF, we designed a motherboard to connect the MEBs. The overall implementation of SMEFF includes 2 parts: hardware and logic design of motherboard. All the memory extension boards (MEBs) are inserted into the motherboard, which forms the extended memory network. The motherboard design includes two aspects: hardware design and logic design.

As shown in Fig.5, the motherboard hardware design mainly includes several modules: 4 local memory slots, 6 memory extension card slots, Test-Stimulus-Generation Module (TSGM), SMEFF M-controller and high speed serial bus network. In this paper, 6 memory extension boards (MEB) are used to form a 3-stage cascade.

The main function of the TSGM is to provide two test benches: STREAM/GUPS and self-testing. STREAM and GUPS are programs used by the HPC Challenge Benchmark (HPCC) to test memory or storage performance. STREAM tests contiguous memory performance, including four simple vector calculations: Copy, Scale, Add, and, Triad. GUPS stresses random access performance. For continuous memory access and random memory access characteristics, the STREAM and GUPS programs run on a Microblaze [37], and generate read/write requests for the memory extension controller (M-contrller). Due to the limited capabilities of the Microblaze, this test cannot fully occupy the bandwidth of the high-speed serial bus, and cannot test the maximum bandwidth of the memory extension system of SMEFF. Therefore, we also performed hardware self-tests which can fully utilize the serial bus bandwidth and attain maximum memory bandwidth.

### B. Implementation of MEB

As shown in Fig. 6, the main hardware modules of each memory extension board (MEB) are as follows : HSSI (high speed serial interface), Router (FPGA), 3 local slots (local memory channels).

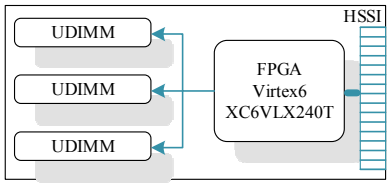


Fig. 6 MEB framework

In terms of cost, the MEB uses low-cost FPGAs. Currently, the cost of these FPGAs is even lower than the price of a UDIMM, which makes this design economically feasible. Inside each MEB, each Virtex 6 FPGA is connected with 3 memory channels. The combination of the FPGA-based and a multi-layer asynchronous memory system provides more than 5x memory capacity gains.

### C. Implementation of Packet

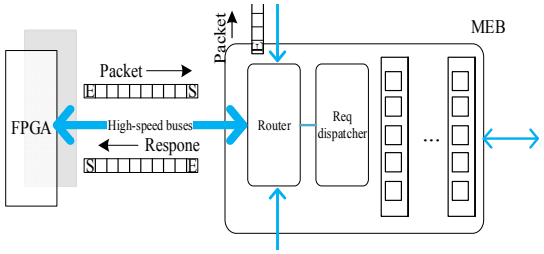


Fig. 7 Packet switching

As shown in Fig. 7, the packets are passed between the MEBs and the FPGAs. A simple routing system is used. Each MEB assigns an ID to its corresponding memory nodes in order to route requests using a packet destination ID. When the destination MEB address of the packets is equal to the ID of the current MEB, the packets are processed at the current MEB. Otherwise, the packets are forwarded to other MEBs by looking up the routing table. A packet can contain multiple read requests and write requests, so that multiple requests can be processed in parallel to the different channels.

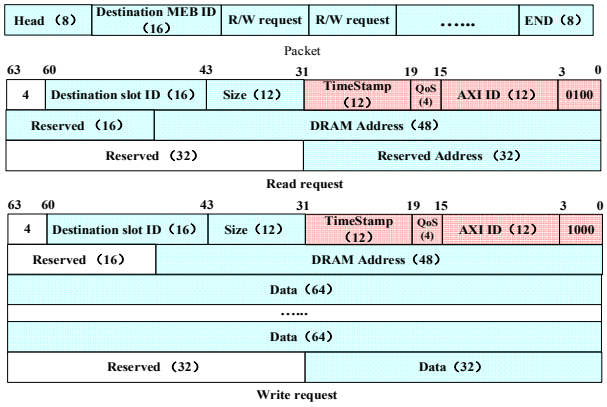


Fig. 8 The implementation of packet

As shown in Fig. 8, A package consists of multiple read / write requests. The packet is sent to the MEB node according to the destination MEB ID. If the MEB ID of the packet is not the MEB ID of this node, it is forwarded.

Time stamping allows packets to be executed out of order. The receiving end rearranges the packets according to the time sequence. Read requests need a handshake response, write requests are not needed.

Read request and Write request process: when the destination channel address of the packets is equal to the ID of the current channel, the packets are processed at the current channel. Otherwise, the packets are forwarded to other channels by looking up the routing table.

Response Packet structure: the response packet structure is the same as the transmitted packet structure, except that the read response returns the data of the read request, a write response returns the ID of the write request.

## V. EVALUATIONS

We evaluate four characteristics: memory capacity, memory bandwidth, latency, and power consumption. Each characteristic is used to emphasize a different benefit of the SMEFF design.

In terms of evaluating the capacity, we selected several practical solutions: FPGA-based Local DRAM, PCIe-based extended memory systems, LRDIMM [25] [38] [39] and HMC.

In terms of memory bandwidth and latency evaluation, we run the STREAM and GUPS programs on a Microblaze [37] to test the bandwidth and latency of each level of SMEFF, and the overall bandwidth and latency. Because Microblaze has limited capability, the bandwidth of the high-speed serial bus cannot be fully occupied, and cannot test the maximum bandwidth of the memory extension system of SMEFF. Therefore, we also performed a self-test which can obtain the maximum memory bandwidth.

In terms of power consumption evaluation, we selected several benchmarks to compare the power consumption ratio between the current FPGA memory controller and the M-TO-M DMA mechanism.

### A. Capacity evaluation

TABLE II. COMPARISON OF THE NUMBER OF CHANNELS SUPPORTED

| | Local DRAM | HMC | LRDIMM | PCIe | SMEFF |
|---|---|---|---|---|---|
| DIMM channel | 4 | 4 | 4 | Unlimited | Unlimited |
| delay | small | small | small | large | small |
| Data movement | High power | High power | High power | High power | Low power |

For Local DRAM, state-of-the-art FPGAs only support 4 channels. For PCIe-based memory systems, the number of cascades is equivalent to SMEFF, but is limited by PCIe protocol restrictions [13]. The latency is high and memory access can only occur in blocks, not more standard 64 byte units. The LRDIMM can supports 4 channels, each of which has a capacity twice that of UDIMMs. Thus, this system offers an equivalent support of 8 UDIMM channels. HMC currently has limited capacity and expensive, the largest HMC is 16G. Obviously, we support DIMMs for 22 channels (18 extension channels and 4 local channels), is 5x capacity gain of the most advanced FPGAs, and 2.5x compared to a LRDIMM solution [25] [40].

### B. Bandwidth and latency evaluation

The hardware transmission delay is tested by inserting the Chipscope ILA [41] module into the communication card. We start a timer when the motherboard sends the first request and stop when the last request returns the data. We can calculate the average delay. The delay in the different levels of SMEFF memory extension system is shown in Tab. 3.

TABLE III. DELAY TESTED BASE ON GUPS ACCESS

| Memory cascade position | Access delay time | Access delay cycles | Aurora Transmission time ratio |
|---|---|---|---|
| FPGA to Local | 0.44 us | 44 | --- |
| FPGA to Level 1 | 1.66us | 166 | 53% |
| FPGA to Level 2 | 3.07us | 307 | 57.3% |
| FPGA to Level 3 | 4.57us | 457 | 57.7% |
| Aurora data delay | 440ns | 44 | --- |

TABLE IV. DELAY TESTED BASE ON STREAM ACCESS

| Memory cascade position | Access delay time | Access delay cycles | Aurora Transmission time ratio |
|---|---|---|---|
| FPGA to Local | 0.44 us | 44 | --- |
| FPGA to Level 1 | 1.26us | 126 | 57% |
| FPGA to Level 2 | 2.66us | 266 | 61.3% |
| FPGA to Level 3 | 4.07us | 407 | 62.7% |
| Aurora data delay | 440ns | 44 | --- |

As shown in Tab. 3 and Tab. 4, we present results for a set of benchmark applications to evaluate the effectiveness of the SMEFF design on existing code.

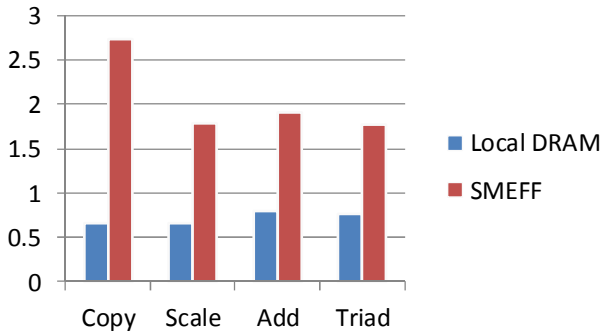


Fig.9 Bandwidth comparison of SMEFF vs. Traditional Memory with STREAM

In the GUPS test, the requested addresses are random, and the amount of requested data is small. Due to the poor locality of requests in this test, the latency is high. In the STREAM

test, the requested addresses are continuous, and the amount of requested data is large. Since the locality of requests is relatively better in this test, the observed latency is about 13% lower than that of the GUPS random memory test, with 4% better bandwidth utilization.

As shown in Fig. 9, the bandwidth of the SMEFF is better than state-of-the-art FPGA memory systems. Particularly in the copy test, the bandwidth of the SMEFF is 3.6 times gain as compared to state-of-the-art FPGA memory systems. Furthermore, the better the locality of requests, the smaller the delay and the higher the bandwidth utilization.

TABLE V. BANDWIDTH TESTED BASE GUPS ON ACCESS

| | Read bandwidth (MB/s) | Write bandwidth (MB/s) |
|---|---|---|
| FPGA to Local | 786 | 1002 |
| FPGA to Level 1 | 488 | 952 |
| FPGA to Level 2 | 480 | 920 |
| FPGA to Level 3 | 480 | 920 |
| total | 2234 | 3794 |

TABLE VI. BANDWIDTH TESTED BASE ON STREAM ACCESS TEST

| | Read bandwidth (MB/s) | Write bandwidth (MB/s) |
|---|---|---|
| FPGA to Local | 806 | 1165 |
| FPGA to Level 1 | 508 | 1141 |
| FPGA to Level 2 | 504 | 1132 |
| FPGA to Level 3 | 506 | 1130 |
| total | 2324 | 4568 |

TABLE VII. MEMORY BANDWIDTH TEST BASE ON HARDWARE SELF-TEST

| | Read bandwidth (MB/s) | Write bandwidth (MB/s) |
|---|---|---|
| FPGA to Local | 5384 | 5481 |
| FPGA to Level 1 | 4750 | 4820 |
| FPGA to Level 2 | 4673 | 4751 |
| FPGA to Level 3 | 4599 | 4710 |
| total | 19406 | 19762 |

The Microblaze is not fast enough [37] [38] to send enough read and write requests to fill the bandwidth of the high-speed serial bus. Thus, we also implemented a hardware self-test to demonstrate the potential overall bandwidth and delay results of SMEFF. As shown in Tab. 6 and 7, SMEFF provides up to 3.6x memory bandwidth improvements and obtains the maximum memory bandwidth by self-test test bench.

### C. *Module to Module DMA copy*

As shown in Fig. 10, we also tested the module-to-module copy performance of SMEFF. We selected four benchmark applications (512M, 1G, 4G, and 8G) to evaluate the effectiveness of the M-TO-M DMA mechanism data movement from one module to another. The latency and power consumption of M-TO-M DMA mechanism is compared to the current FPGA memory system. We find that, as data volumes increase, the M-TO-M DMA mechanism is superior to power ratio and latency ratio.

As shown in Tab. 8 and Fig. 10, the data movement of M-TO-M's DMA technology provides about 3x improvement in terms of latency compared to state-of-the-art FPGA-based memory systems. We find that as the copy scales larger, the M-TO-M delay advantages also grow.

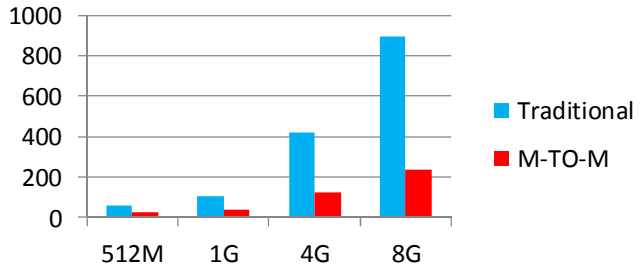


Fig.10 Different size data movement time comparison delay percentage

TABLE VIII. DELAY (TRADITIONAL / M-TO-M)

| | 512M | 1G | 4G | 8G |
|---|---|---|---|---|
| Tr./ M-to-M | 2.3 | 2.81 | 3.4 | 3.8 |

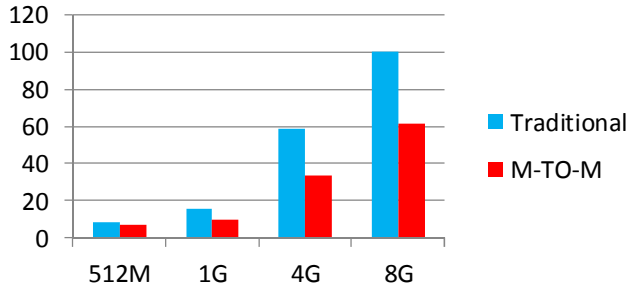


Fig.11 Comparison of energy consumption

TABLE IX. REDUCTION PERCENTAGE OF ENERGY CONSUMPTION

| | 512M | 1G | 4G | 8G |
|---|---|---|---|---|
| M-to-M/ Tr. | 21.1% | 29% | 43.3% | 61.1% |

As shown in Tab. 9 and Fig. 11, the data movement of M-TO-M's DMA technology provides a 21.1% to 61.1% reduction in power consumption compared to state-of-the-art FPGA-based memory systems. We find that as the data scales larger, energy reduction radio of the M-TO-M DMA technology also grow.

## VI. CONCLUSION AND FUTURE WORK

In terms of memory capacity and bandwidth, we present a new design memory organization, SMEFF. We exploit FPGA-based high-speed serial bus to build a memory network. SMEFF provides 5x memory capacity gains and up to 3.6x memory bandwidth improvements compared to state-of-the-art FPGA's memory systems. It is also superior to PCIe-based extended memory systems. SMEFF introduces the concept of a multi-level and scalable asynchronous memory system. In terms of memory latency and power consumption, the data movement of M-TO-M's DMA technology achieves up to 3x

latency reduction, and energy reduction by 21.1% to 61.1% on average compared to state-of-the-art FPGA-based memory systems.

Our work enables many new research opportunities. We plan to make the SMEFF even more efficient by applying optimizations as prefetching, providing a flexible (vs. cache line based) communication granularity. More broadly, we would like to expand the SMEFF idea to other specialized applications in the context of other compute units.

## ACKNOWLEDGEMENTS

We would like to thank shepherd and the anonymous reviewers for their insightful comments and suggestions. This work is supported by National Key Research and Development Plan of China 2017YFB1001600, NSFC (National Science Foundation of China) No. 61331008, 61521092 and 61772497, State Key Laboratory of Computer Architecture foundation under grant CARCH2601 and HuaweiYBCB2011030. Corresponding author is Yuhang Liu.

## REFERENCES

[1] Xilinx, "UltraScale Architecture-Based FPGAs Memory IP v1.4 data sheet," 2017.

[2] "VMware to increase consolidation ratio to 16 VMs/ core " http://virtualization.info/en/news/2010/01/vmware-to-increase-consolidation-ratio.html.

[3] YuHang Liu and Xian-He Sun, "C2-bound: a capacity and concurrency driven analytical model for many-core design," International Conference for High Performance Computing, Networking, Storage and Analysis(SC), pages 1-11, 2015.

[4] YuHang Liu and Xian-He Sun, "LPM: Concurrency-Driven Layered Performance Matching," International Conference on Parallel Processing, pages 879-888, 2015.

[5] YuHang Liu and Xian-He Sun, "Reevaluating Data Stall Time with the Consideration of Data Access Concurrency," Journal of Computer Science and Technology, 30(2):27-245, 2015.

[6] Cisco, "Cisco unified computing system extended memory technology overview," http://www.cisco.com/c/en/us/products/collateral/servers-unified-computing/ucs-b-series-blade-servers/white_paper_c11-525300.html.

[7] E. Cooper-Balis, P. Rosenfeld, and B. Jacob, "Buffer-on-board memory systems," in Proceedings of the 39th Annual International Symposium on Computer Architecture, pages 25-27, 2012.

[8] Micron, "Rack up server performance with load-reduced DIMMs,"http://www.micron.com/products/dram-modules/lrdimm.

[9] Hybrid Memory Cube Consortium, "Hybrid Memory Cube Specification1.0," 2013.

[10] A. N. Udipi, N. Muralimanohar, R. Balasubramonian, A. Davis, and N. P. Jouppi, "Combining memory and a controller with photonics through 3d-stacking to enable scalable and energy-efficient systems," in ACM SIGARCH Computer Architecture News, 39(3):425-436, 2011.

[11] K. Lim, J. Chang, T. Mudge, P. Ranganathan, S. K. Reinhardt, and T. F. Wenisch, "Disaggregated memory for extension and sharing in blade servers," in Proceedings of the 36th Annual International Symposium on Computer Architecture, 37(3):267-278, 2009.

[12] K. Lim, Y. Turner, J. R. Santos, A. AuYoung, J. Chang, P. Ranganathan, and T. F. Wenisch, "System-level implications of disaggregated memory," in 2012 IEEE 18th International Symposium on High Performance Computer Architecture (HPCA), 8(3):1-12, 2012.

[13] R. Hou, T. Jiang, L. Zhang, P. Qi, J. Dong, H. Wang, X. Gu, andS. Zhang, "Cost effective data center servers," in 2013 IEEE 19th International Symposium on High Performance Computer Architecture (HPCA), pp. 179-187, 2013.

[14] Udit Dhawan, André DeHon, "Area-efficient near-associative memories on FPGAs," FPGA, 7(4):191-200, 2013.

[15] Hybrid Memory Cube Consortium, "Hybrid Memory Cube Specification1.0," 2013.

[16] Hybrid Memory Cube Consortium, "Hybrid Memory Cube Specification2.1," 2015.

[17] [ "HBM2," http://www.samsung.com/semiconductor/products/dram/-graphic-dram.

[18] Intel. Intel 7500/7510/7512 scalable memory buffer datasheet, 2011.

[19] Cisco, "Cisco unified computing system extended memory technology overview," http://www.cisco.com/c/en/us/products/collateral/servers-unified-computing/ucs-b-series-blade-servers/white_paper_c11-525300.html.

[20] JEDEC, "JESD206: FBDIMM architecture and protocol," 2007.

[21] JEDEC, "JESD79-3E: DDR3 SDRAM specification," 2010.

[22] JEDEC, "JESD79-4: DDR4 SDRAM specification," 2012.

[23] JEDEC, "JESD209-2F: LOW POWER DOUBLE DATA RATE 2 (LPDDR2)," 2013.

[24] ITRS, "ITRS report 2012 update," 2012.

[25] Intel, "DDR3 LRDIMM system-level validation results on intel xeone5-2600 v2 processor family," 2014.

[26] Nathaniel Mcvicar, Chih-Ching Lin and Scott Hauck, "K-mer Counting Using Bloom Filters with an FPGA-Attached HMC," the 25th IEEE International Symposium on Field-Programmable Custom Computing Machines(FCCM), pages 203-210, 2017.

[27] Muhsen Owaida, David Sidler, Kaan Kara and Gustavo Alonso, "Centaur: A Framework for Hybrid CPU-FPGA Databases," the 25th IEEE International Symposium on Field-Programmable Custom Computing Machines(FCCM), pages 211-218, 2017.

[28] JEDEC. JESD82-20A: FBDIMM advanced memory bu_er (AMB), 2009.

[29] B. C. Lee, E. Ipek, O. Mutlu, and D. Burger, "Architecting phase change memory as a scalable dram alternative," in Proceedings of the 36th Annual International Symposium on Computer Architecture(ISCA), 37(3):2-13, 2009.

[30] Intel, "Mobile LPDDR2-PCM 512mb, 1gb, x16, 1.8v/1.2v i/o 45nm phase change memory," 2012.

[31] "ULLtraDIMM," http://www.sandisk.com/enterprise/ulltradimm-ssd/.

[32] L.-C. Chen, M.-Y. Chen, Y. Ruan, Y.-B. Huang, Z.-H. Cui, T.-Y. Lu, and Y.-G. Bao, "MIMS: Towards a message interface based memory system," Journal of Computer Science and Technology, vol. 29, no. 2, pp. 255–272, 2014.

[33] "Virtex UltraScale +" https://www.xilinx.com/products/silicon-devices/-fpga/virtex-ultrascale-plus.html

[34] Xilinx, "UltraScale Architecture GTH Transceivers" 2017

[35] "Virtex 7" https://www.xilinx.com/search/site-keyword-search.html-searchKeywords=Virtex7

[36] "Virtex 6," https://www.xilinx.com/products/.html

[37] "Microblaze" https://www.xilinx.com/products/design-tools.-html

[38] Intel, "DDR3 RDIMM system-level validation results on intel xeon e5-2600 v2 processor family," 2014.

[39] Intel, "Intel 64 and IA-32 architectures software developer's manual,"2014.

[40] Edin Kadric, David Lakata, André DeHon, "Impact of Memory Architecture on FPGA Energy Consumption," FPGA, pages 146-155, 2015.

[41] "chipscope ILA" https://www.xilinx.com/products/intellectual-property/chipscope_icon.htm