RDMA over Converged Ethernet

Sherif Badran, Rami Rasheedi, Zeyad Madbouly, et al.
Computer and Systems Department
Faculty of Engineering, Ain Shams University
Cairo, Egypt

July 23, 2021

Ain Shams University, Faculty of Engineering

Electronics and Communications Engineering Department

Cairo, Egypt

RDMA over Converged Ethernet

A Report Submitted in Partial Fulfillment of the Requirements of the Degree of Bachelor of Science in Electronics and Communications Engineering

By

Mohammed Hussien Mostafa	**1601160**
Mohamed khaled Mohamed Sayed El khawas	**1601173**
Christine Magdy Gad El-Rab Samuel	**16E0129**
Martina Fadi Fouad farag	**1601053**
Maryham melad Gerges Michael	**1601075**

Supervised by

Prof. Dr. Victor Goraldiz

Cairo 2021

Declaration

We hereby certify that this project submitted as part of our partial fulfillment of BSc in Electronics and Communications Engineering is entirely our own work, that we have exercised reasonable care to ensure its originality, and does not to the best of our knowledge breach any copyrighted materials, and have not been taken from the work of others and to the extent that such work has been cited and acknowledged within the text of our work.

Signed

Name	Signature
Mohammed Hussien Mostafa	
Mohamed khaled Mohamed Sayed El khawas	
Christine Magdy Gad El-Rab Samuel	
Martina Fadi Fouad farag	
Maryham melad Gerges Michael	

Date: Day of 24$^{\text{th}}$ of July, in the year 2021.

Acknowledgments

Thank and acknowledge your advisor, family and friends.

Abstract

Nowadays, data have become a crucial aspect of all evolving computer technologies. That's why owning data and knowing how to use it efficiently has become a key of success to any enterprise. Not only is data a crucial aspect of our modern life, but also the speed at which applications are running and data can be accessed has a significant impact as well. However, massive amounts of data that need to be analyzed, processed, shared, transferred, monitored and accessed (Big data/parallel computing/Cloud computing) have led to very high work load on data centers and systems. This has led to slower processing speeds and increased latency in response to user requests or operations running. In addition, the high availability requirement of any database has become exceedingly important and backup systems are taken into consideration from the very beginning at the phase of designing the system to ensure no or minimum down time and continuous functionality.

Our project proposes a methodology to tackle the above problems where applications can directly access the memory and perform I/O operations without CPU interference through DMA directly. Hence, minimizing latency and maximizing the CPU processing speed.

RDMA allows direct access of memory of one computer into the other's memory without involving any OS. This is very useful nowadays in the distributed systems where individual computers are connected together and are communicating together easily to facilitate efficient data transfer and parallel processing and resource sharing to appear as one integrated system.

There are various RDMA protocols such as RoCE V.1 and RoCE V.2 and IWARP. Ethernet is an alternative RDMA offering that is more complex and unable to achieve the same level of performance as RoCE-based solutions.

However, in our project, we have implemented RoCE V.1. RoCE is a network protocol which allows accessing memory directly over an Ethernet network. This is done through the encapsulation of IB transport packet over Ethernet.

Contents

1 Introduction **1**
- 1.1 Introduction . . . 1
- 1.2 Problem Statement . . . 1
- 1.3 Objective . . . 1
- 1.4 Outline . . . 1

2 Introduction to Virtual PCIe **2**

3 Linux Kernel **3**

4 Introduction to Memory systems **4**

5 DDR5 Memory New Features **5**

6 DDR5 Controller Architecture **6**
- 6.1 Front-end . . . 6
- 6.2 Back-end . . . 6

7 DDR5 Controller Specifications **7**
- 7.1 Address Mapping Scheme . . . 7
- 7.2 Memory requests scheduling . . . 7
  - 7.2.1 FCFS . . . 7
- 7.3 Bank Arbitration . . . 7
  - 7.3.1 Idea Behind Arbiter Design . . . 7
  - 7.3.2 Block Diagram . . . 8

8 Emulation Results [1, 2] [3] **9**

9 Conclusions and Future Work **10**

Bibliography **11**

A First Appendix **12**

B Second Appendix **13**

List of Figures

7.1 Arbiter block diagram 8

List of Tables

List of Algorithms

1.1 Anything 1

List of Abbreviations

Abbreviations

API	Apllication Programming Interface
CPU	Central Processing Unit
CQE	Completion Queue Element
DDR	Double Data Rate
DRAM	Dynamic Random Access Memory
FCFS	First come, First served
FR-FCFS	First ready-First come, First served
FSM	Finite State Machine
HCA	Host Channel Adapter
HPC	High Performance Computing
HWM	High watermark
IETF	Internet Engineering Task Force
iWARP	Internet wide-Area Network
LWM	Low watermark
NIC	Network Interface Card
OS	Operating System
QP	Queue Pair
RDMA	Remote Direct Memory Access
RLDP	Row Locality based Drain Plicy
RoCE	RDMA over Converged Ethernet
RoCE V.1	RDMA over Converged Ethernet version 1
RoCE V.2	Routable RoCE
WOE	Work Queue Element

Chapter 1

Introduction

1.1 Introduction

1.2 Problem Statement

1.3 Objective

1.4 Outline

Algorithm 1.1 Anything

Require: $\rho \geq 1$
Ensure: X_k
1: **while** not converged **do**
2: Solve $X_{k+1} = \min_X L(X, Y_k, \mu_k)$
3: $Y_{k+1} = Y_k + \mu_k h(X_{k+1})$
4: $\mu_{k+1} = \rho \mu_k$
5: **end while**

Chapter 2

Introduction to Virtual PCIe

Chapter 3

Linux Kernel

Chapter 4

Introduction to Memory systems

Chapter 5

DDR5 Memory New Features

Chapter 6

DDR5 Controller Architecture

6.1 Front-end

6.2 Back-end

Chapter 7

DDR5 Controller Specifications

7.1 Address Mapping Scheme

7.2 Memory requests scheduling

DRAM is the most commonly used technology for building memory systems. However, it has been a main performance bottleneck for modern computer systems. Hence, many request scheduling algorithms are designed in order to reduce latency and exploit maximum row buffer locality. Exploiting row buffer locality in DRAM is a main key characteristic while designing proper scheduling algorithm for application needs. DRAM architecture is segmented into multiple banks to support concurrent accesses. Each DRAM bank consists of rows and columns of DRAM cells. Each bank is accessible with a row buffer, accessing it is faster than accessing different row in the same bank [4].

We will discuss the main scheduling algorithms we studied during designing phase:

1. FCFS
2. FR-FCFS
3. RLDP
4. Thread-Fair Request Reordering

7.2.1 FCFS

7.3 Bank Arbitration

7.3.1 Idea Behind Arbiter Design

Before we discuss in such a deep way into design, we should show some important timing constraints in the new DDR5 technology. First,

Our proposed controller arbitration criteria aims to exploit maximum concurrent processess supported by DDR5 bank groups technology, thus, our arbiter is aiming to grant access to controller data path to new bank groups if there are ready requests in it, Hence, exploiting maximum available concurrent accessing into a DDR5 chip.

Choosing suitable arbitration sequence is so critical in order to increase or performance.

There are three timing constrains we should take into consideration in the new DDR5 technology:

- Short access timing between two different bank groups:

Arbiter must take in consideration this constrain, whatever the current group is, arbiter should select new different group to drain from.

- Long access timing between two different banks in the same bank group and worst access timing between different rows in same bank:

Arbiter should choose different banks in same group before trying to give access to same last bank in order to avoid row conflicts as possible.

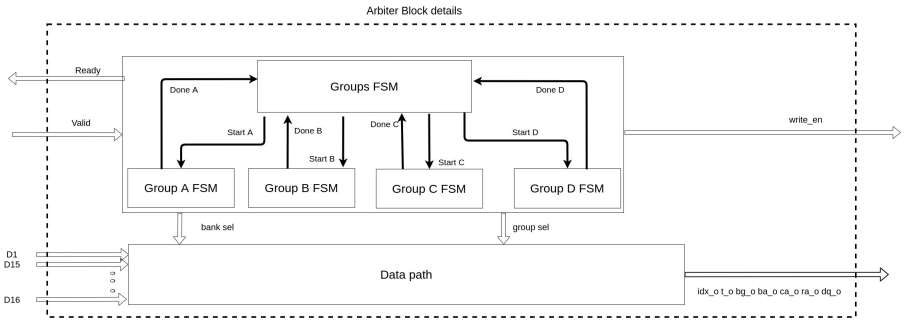

Figure 7.1: Arbiter block diagram

7.3.2 Block Diagram

hi iam block diagram

Chapter 8

Emulation Results [1, 2] [3]

edgdf		gfgf			fgfgfg
dfgd		dfgf			fgdggfgdg
fgf		gfgf			gfgf
		gf	fg		gf
fgf		gfg			

Chapter 9

Conclusions and Future Work

Bibliography

[1] B. Jacob, D. Wang, and S. Ng, *Memory systems: cache, DRAM, disk*. Morgan Kaufmann, 2010.

[2] D. J. Smith, *HDL Chip Design: A practical guide for designing, synthesizing and simulating ASICs and FPGAs using VHDL or Verilog*. Doone publications, 1998.

[3] W. Mauerer, *Professional Linux kernel architecture*. John Wiley & Sons, 2010.

[4] Y.-S. Moon, Y. Kwon, H.-S. Kim, D.-g. Kim, H. H. Lee, and K. Park, "The compact memory scheduling maximizing row buffer locality," in *3rd JILP Workshop on Computer Architecture Competitions: Memory Scheduling Championship, MSC*, 2012.

Appendix A

First Appendix

Appendix B

Second Appendix