Diplomarbeit

Data Movement in Heterogeneous Memories with Intel Data Streaming Accelerator

Anatol Constantin Fürst

7th January 2024

Technische Universität Dresden
Fakultät Informatik
Institut für Systemarchitektur
Professur Betriebssysteme

Betreuender Hochschullehrer:	Prof. Dr.-Ing. Horst Schirmeier
Betreuender Mitarbeiter:	M.Sc. André Berthold

TECHNISCHE
UNIVERSITÄT
DRESDEN

Fakultät Informatik Institut für Systemarchitektur, Professur für Betriebssysteme

Aufgabenstellung für die Anfertigung einer Bachelor-Arbeit

Studiengang:	Bachelor
Studienrichtung:	Informatik (2009)
Name:	Constantin Fürst
Matrikelnummer:	4929314
Titel:	Data Movement in Heterogeneous Memories with Intel Data Streaming Accelerator

Developments in main memory technologies like Non-Volatile RAM (NVRAM), High Bandwidth Memory (HBM), NUMA, or Remote Memory, lead to heterogeneous memory systems that, instead of providing one monolithic main memory, deploy multiple memory devices with different non-functional memory properties. To reach optimal performance on such systems, it becomes increasingly important to move data, ahead of time, to the memory device with non-functional properties tailored for the intended workload, making data movement operations increasingly important for data intensive applications. Unfortunately, while copying, the CPU is mostly busy with waiting for the main memory, and cannot work on other computations. To tackle this problem Intel implements the Intel Data Streaming Accelerator (Intel DSA), an engine to explicitly offload data movement operations from the CPU, in their newly released Intel Xeon CPU Max processors.

The goal of this bachelor thesis is to analyze and characterize the architecture of the Intel DSA and the vendor-provided APIs. The student should benchmark the performance of Intel DSA and compare it to the CPU's performance, concentrating on data transfers between DDR5-DRAM and HBM and between different NUMA nodes. Additionally, the student should find out in what way and to what extent parallel processes copying data interfere with each other. Analyzing the performance information, the thesis should outline a gainful utilization of the Intel DSA and demonstrate its potential by extending the Query-driven Prefetching concept, which aims to speed up database query execution in heterogeneous memory systems.

Gutachter:	Prof. Dr.-Ing. Dirk Habich
Betreuer:	André Berthold, M.Sc.
Ausgehändigt am:	24. November 2023
Einzureichen am:	9. Februar 2024

Prof. Dr.-Ing. Horst Schirmeier
Betreuender Hochschullehrer

Selbständigkeitserklärung

Hiermit erkläre ich, dass ich diese Arbeit selbstständig erstellt und keine anderen als die angegebenen Hilfsmittel benutzt habe.

Dresden, den 7. Januar 2024

Anatol Constantin Fürst

Abstract

...abstract ...

write abstract

Contents

Todo list

List of Figures

List of Tables

1 Introduction

adopt title page

adopt disclaimer

write introduction

1.1 A Section

Referencing other chapters: 2 3 4 5 6 7

Name	**Y**	**Z**
Foo	20,614	23 %
Bar	9,914	11 %
Foo + Bar	30,528	34 %
total	88,215	100 %

Table 1.1: Various very important looking numbers and sums.

More text referencing Table 1.1.

1.2 Another Section

Citing [Bel05] other documents [Bel05; Boi06] and Figure 1.1.

Something with umlauts and a year/month date: [BD04].

And some online resources: [Gre04], [Hub89]

1.3 Yet Another Section

add content

1.4 Test commands

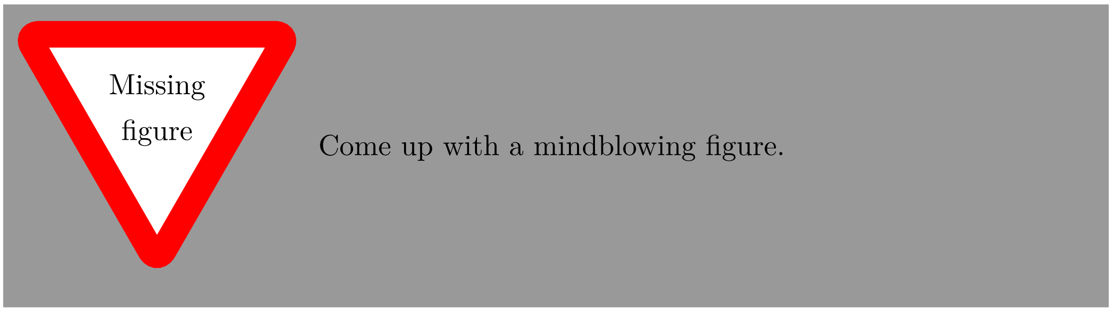
Drops L^{4}LinuxNOVA QEMU `memcpy` A sentence about BASIC. And a correctly formatted one about ECC.

1.5 Test Special Chars

Before you start writing your thesis please make sure that your build setup compiles the following special chars correctly into the PDF! If for example ß is printed as 'SS' then you should fix this! There are a few hints in the repository in `preamble/packages.txt`.

ö ä ü Ö Ä Ü ß < >

Figure 1.1: A long description of this squirrel figure. Image taken from `http://commons.wikimedia.org/wiki/File:Sciurus-vulgaris_hernandeangelis_stockholm_2008-06-04.jpg`



Come up with a mindblowing figure.

Figure 1.2: A mindblowing figure

2 Technical Background on Intel DSA

> Intel DSA is a high-performance data copy and transformation accelerator that will be integrated in future Intel® processors, targeted for optimizing streaming data movement and transformation operations common with applications for high-performance storage, networking, persistent memory, and various data processing applications. [Cor22a, p. 15]

Introduced with the 4th generation of Intel Xeon Scalable Processors [Cor22b], the DSA promises to alleviate the CPU from ‘common storage functions and operations such as data integrity checks and deduplication’ [Cor22b]. This chapter will give an overview of the architecture, software and the interaction of these two components. The reader will be familiarized with the setup and equipped with the knowledge to configure the system for a specific use case.

consider adding projected use cases as in the architecture specification here

2.1 Architecture

To be able to optimally utilize the Hardware, knowledge of its workings is required to make educated decisions. Therefore, this section describes both the workings of the DSA engine itself (referred to as internal architecture) and the way it integrates with the rest of the processor (external architecture). All statements are based on Chapter 3 of the Architecture Specification by Intel [Cor22a].

As the accelerator is directly integrated into the CPU, a system with multiple processors, as it is common in servers, will also have multiple DSAs. These engines are accessible via the CPUs IO-Fabric as a PCIe device, and submit memory requests through this BUS directly to the Input/Output Memory Management Unit (IOMMU). Configuration of the device on a low level is done through memory-mapped I/O registers that are set in the Base Address Register (BAR), which is also used to set the location of work submission portals. Through these portals, the so-called work descriptors are handed over to the device for processing.

- possibly more performance with multiple engines per group (and single WQ) to cover over high latency address translation [Cor22a, p. 25]
- drain descriptor / drain command signals completion of preceding descriptors for fencing in non-batch submissions, in batches the “fence flag“ can be used to ensure ordering, failures before a fence will lead to the following descriptors being aborted [Cor22a, p. 30], `sfence` or `mfence` should be executed before pushing drain descriptor [Cor22a, p. 32]

- cache control flag in descriptor controls whether writes are directed to cache or to memory [Cor22a, p. 31] effects on copy from DRAM > HBM unknown
- shared WQ receive work via 'PCIe deferrable memory write request' to the portal which removes the need for synchronization of submissions but can cost more due to the communication overhead of posting a write request and waiting for it to be signalled 'completed' [Cor22a, p. 23]
- dedicated WQ are configured by the driver with a specified PASID for address translation and can not be shared by multiple clients [Cor22a, p. 24]

2.2 HW/SW Setup

Give the reader the tools to replicate the setup. Also explain why the BIOS-configs are required.

Setup Requirements:

- VT-d enabled
- limit CPUPA to 46 Bits disabled
- IOMMU enabled
- kernel with iommu and DSA support
- kernel option "intel_iommu=on,sm_on"

Software Configuration: Describe intel accel-config and how it works with back reference to architecture.

Software Access: Explain how a piece of software may access the DSA/WQ, how the drivers and dsa libraries enable this and also how access policies are enforced.

2.3 Microbenchmarks

provide microbenchmarks with multiple configurations and for many use cases

2.4 Evaluation

evaluate the benchmarks and conclude with projected use cases - may use the cases from dsaspec/guide

3 Design

3.1 Introduction VAMPIR

- Hardware Overview with CPU/RAM/HBM/NUMA-Nodes in Graph
- Overview of Software with querry-pipeline

3.2 Analysis of Applicability of DSA

- Benchmark the amount of time spent on memory operations in VAMPIR
- Back-reference to the Microbenchmarks and conclusion on possible gains

4 Implementation

...implementation ...

write implementation

5 Evaluation

...evaluation ...

write evaluation

6 Future Work

...future work ...

write future work

7 Conclusion And Outlook

...conclusion ...

write conclusion

Glossary

B

BAR

... desc ...

D

DSA

... desc ...

I

IOMMU

... desc ...

Bibliography

[BD04] Michael Becher and Maximillian Dornseif. 'Feuriges Hacken - Spaß mit Firewire'. In: *21C3: Proceedings of the 21st Chaos Communication Congress.* Dec. 2004.

[Bel05] Fabrice Bellard. 'QEMU, a fast and portable dynamic translator'. In: *Proceedings of the USENIX Annual Technical Conference, FREENIX Track.* 2005, pp. 41–46.

[Boi06] Adam Boileau. 'Hit by a Bus: Physical Access Attacks with Firewire'. In: *RUXCON.* 2006.

[Cor22a] Intel Corporation. *Intel® Data Streaming Accelerator Architecture Specification.* 16th Sept. 2022. URL: `https://www.intel.com/content/www/us/en/content-details/671116/intel-data-streaming-accelerator-architecture-specification.html` (visited on 15th Nov. 2023).

[Cor22b] Intel Corporation. *New Intel® Xeon® Platform Includes Built-In Accelerators for Encryption, Compression, and Data Movement.* Dec. 2022. URL: `https://www.intel.com/content/dam/www/central-libraries/us/en/documents/2022-12/storage-engines-4th-gen-xeon-brief.pdf` (visited on 15th Nov. 2023).

[Gre04] Tom Green. *1394 Kernel Debugging Tips and Tricks.* Slide presentation at the WinHEC 2004. 2004. URL: `http://download.microsoft.com/download/1/8/f/18f8cee2-0b64-41f2-893d-a6f2295b40c8/DW04001_WINHEC2004.ppt` (visited on 3rd June 2009).

[Hub89] William S. Huber. 'Operating system debugger'. 4819234 (Needham, MA). Apr. 1989. URL: `http://www.freepatentsonline.com/4819234.html`.