TU Delft

Delft University of Technology

On Leveraging Vertical Proximity in 3D Memory Hierarchies

Lefter, Mihai

DOI
10.4233/uuid:f744c1af-505e-440c-bc49-2a1d95d0591d

Publication date
2018

Document Version
Final published version

Citation (APA)
Lefter, M. (2018). *On Leveraging Vertical Proximity in 3D Memory Hierarchies*. [Dissertation (TU Delft), Delft University of Technology]. https://doi.org/10.4233/uuid:f744c1af-505e-440c-bc49-2a1d95d0591d

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Copyright
Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

Takedown policy
Please contact us and provide details if you believe this document breaches copyrights.
We will remove access to the work immediately and investigate your claim.

This work is downloaded from Delft University of Technology.
For technical reasons the number of authors shown on this cover page is limited to a maximum of 10.

On Leveraging Vertical Proximity in 3D Memory Hierarchies

Cover inspired by the works of Dirk Huizer and Anatoly Konenko.

On Leveraging Vertical Proximity in 3D Memory Hierarchies

Dissertation

for the purpose of obtaining the degree of doctor
at Delft University of Technology
by the authority of the Rector Magnificus Prof. dr. ir. T.H.J.J. van der Hagen
chair of the Board for Doctorates
to be defended publicly on

Wednesday 14 November 2018 at 10:00 o'clock

by

Mihai LEFTER

Master of Science in Computer Engineering
Delft University of Technology, The Netherlands
born in Brașov, Romania

This dissertation has been approved by the promotors.

Composition of the doctoral committee:

Rector Magnificus, voorzitter	Delft University of Technology
Dr. S.D. Coțofană	Delft University of Technology, promotor
Dr. J.S.S.M. Wong	Delft University of Technology, copromotor

Independent members:

Prof. dr. P.J. French	Delft University of Technology
Prof. dr. J. Pineda de Gyvez	Eindhoven University of Technology, The Netherlands
Prof. dr. A. Rubio Solá	Polytechnic University of Catalonia, Spain
Prof. dr. L. Vințan	Lucian Blaga University of Sibiu, Romania
Prof. dr. L. Anghel	University Grenoble-Alpes, France
Prof. dr. W. A. Serdijn	Delft University of Technology, reserve member

ISBN 978-94-6186-983-8

Keywords: 3D stacked integrated circuits, nems, nemfet, zero-energy, memory hierarchy, reliability.

Copyright © 2018 Mihai Lefter
All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without permission of the author.

Printed in The Netherlands.

Dedicated to my family and friends

Abstract

Within the past half century, Integrated Circuits (ICs) experienced an aggressive, performance driven, technology feature size scaling. As the technology scaled into the deep nanometer range, physical and quantum mechanical effects that were previously irrelevant become influential, or even dominant, resulting in, e.g., not any longer negligible leakage currents. When attempting to pattern such small-geometry dimensions, the variability of technological parameters considerably gained importance. Furthermore, it became more difficult to reliably handle and integrate such a huge number of tiny transistors into large scale ICs, considering also that a substantial increase in power density needed to be taken into account. Scaling induced performance was no longer sufficient for delivering the expected improvements, which lead to a paradigm switch from uniprocessors to multiprocessor micro-architectures. At the same time, since for certain application domains, such as big data and Internet of things, the to be processed data amount increases substantially, computing system designers become more concerned with ensuring data availability than with reducing functional units latency. As a result, state of the art computing systems employ complex memory hierarchies, consisting of up to four cache levels with multiple shared scenarios, making memory a dominant design element that considerably influences the overall system performance and correct behavior. In this context, 3D Stacked Integrated Circuit (3D SIC) technology emerges as a promising avenue in enabling new design opportunities since it provides the means to interconnect devices with short vertical wires. In this thesis we address the above mentioned memory challenges by investigating the 3D SIC technology utilization in memory designs, as follows. First, we propose a novel banked multi-port polyhedral memory that provides an enriched access mechanism set with a very low bank conflict rate and we evaluate its potential in shared caches. Second, we propose a low power hybrid memory in which 3D technology allows for the smooth co-integration of: (i) short circuit current free Nano-Electro-Mechanical Field Effect Transistor (NEMFET) based inverters for data storage, and, (ii) CMOS-based logic for read/write operations and data preservation. Third, we propose a memory repair framework that exploits the 3D vertical proximity for inter-die redundant resources sharing. Finally, we propose novel schemes for performing user transparent multi-error correction and detection, with the same or even lower redundancy than the one required by state of the art extended Hamming single error correction schemes.

Acknowledgments

I would like to express my gratitude to Dr. Sorin Coțofană, my promotor and daily supervisor. Sorin did not only grant me the opportunity to pursue a PhD, but he also guided and challenged me during this almost everlasting journey. He sacrificed many late evenings, weekends, and holidays to help me with my research. He always encouraged me to develop myself, and his deep insights were very valuable to my work. Over the years, his friendly attitude, inspiring ideas, and unbounded optimism gave me the strength to not lose focus and to always move forward. In addition, thank you Sorin for the delicious dinner events that sparkled a lot of fruitful debates and open discussions that shaped my views on many issues.

Thank you George, Saleh, and Marius, my project and office colleagues, for creating the greenest faculty environment in our office, which enhanced such a wonderful working atmosphere. In addition to the productive work discussions, I deeply enjoyed your friendship and our countless debates on world matters. Mottaqiallah, thank you for the interesting talks we had during and outside the 3DIM^3 project, for your motivation, and for translating the thesis abstract into Dutch. I would also like to thank the co-authors of the publications related to this thesis: Prof. Dr. Said Hamdioui, Dr. Demid Borodin, Dr. Thomas Marconi, and Dr. Valentin Savin. Also, thank you Demid for stirring the photography passion in me, and thank you Thomas for your great introduction into the LDPC world.

Special thanks to Lidwina for her assistance in smoothing all the bureaucratic matters and for the fun discussions that we had, as well as to Erik and Eef, who always did their best to ensure that all of our computer-related problems were solved as quickly as possible. I would also like to thank all colleagues from the CE department for providing an excellent and inspiring working atmosphere.

My staying in The Netherlands would have not been so enjoyable if not for the great Romanian community. I wish to use this opportunity to thank Bogdan, Adriana, Madalina, Marina, Ana Andreea, Alex Catalin, Razvan, Roxana, Alex, Sorina, Dragos, Remus, Alin, Radu, Elena, Vlad, Razvan, Tina, Ozana, Anca, Nicoleta, Andrei, Nicolae Emanuel, Raluca, Cristi, Iulia, Bogdan, Ruxandra, Mihai, and all the others I may have missed. In addition, in Delft I made many great international friends. Thank you Changlin, Yao, Chunyang, Asha, Peter, Olga, Mafalda, Imran, Laiq, Faisal, Seyab, Innocent, Andre, and all the other people in the CE group.

I do not want to miss the occasion to thank Bogdan, Catalin, and Prof. Dr. Georgi Gaydadjiev who ensured that I had to opportunity to study at TU Delft.

I am especially grateful to Jeroen Laros for the confidence boost and support required to continue and finish this PhD.

Last but not least, I wish to thank my family. Mom and dad, thanks for always encouraging me, for your love and continuously warm support. My sister Elena and my brother-in-law Ciprian, I am grateful for your emboldening and affection. My parents in law, Cristina and Iustin, thank you for your encouragements. In addition, I am very fortunate to have a son as Matei - thank you for making my days so happy through your sparkling laughter and joy. And most of all I want to thank Iulia for always being there for me.

Mihai Lefter — Delft, The Netherlands, October 2018

Table of Contents

Title . . . . **iii**

Abstract . . . . **vii**

Acknowledgments . . . . **ix**

List of Tables . . . . **xv**

List of Figures . . . . **xvii**

1 Introduction **1**

1.1 Modern Processor Design Challenges 8

1.1.1 Technology Scaling 10

1.1.2 Ensuring Data Availability in Multi-core Processors 14

1.1.3 Processor Design Within a Restricted Power Envelope . . . 15

1.1.4 Memory Dependability in the Nano Scale Era 17

1.2 Technology Enablers 18

1.2.1 3D Integration Technology 18

1.2.2 Emerging Technologies 25

1.3 Research Questions 27

1.4 Dissertation Contributions 31

1.5 Dissertation Organization 35

Bibliography 37

2 A Shared Polyhedral Cache for 3D Wide-I/O Multi-core Computing Platforms **51**

2.1 Introduction 52
2.2 Wide-I/O Shared Polyhedral Cache 54
2.2.1 Traditional Cache Design 54
2.2.2 Polyhedral Cache Design 55
2.3 Experimental Evaluation 57
2.4 Conclusion 61
Bibliography 63

3 Energy Effective 3D Stacked Hybrid NEMFET-CMOS Caches 65
3.1 Introduction 66
3.2 Low Power Short Circuit Current Free NEMFET Inverter 67
3.3 Low Leakage 3D-Stacked Hybrid NEMFET-CMOS Caches 69
3.3.1 3D-Stacked Hybrid NEMFET-CMOS Memory 69
3.3.2 3D-Stacked Hybrid NEMFET-CMOS Caches 72
3.4 System Level Evaluation 74
3.4.1 Evaluation Methodology and Metrics 75
3.4.2 Analysis 77
3.5 Conclusion 78
Bibliography 81

4 Is TSV-based 3D Integration Suitable for Inter-die Memory Repair? . . . 83
4.1 Introduction 84
4.2 Redundancy Based Memory Repair 85
4.2.1 Previous Work in 3D Memory Repair 87
4.3 3D Inter-die Memory Repair Architecture 88
4.4 3D Inter-die Memory Repair Infrastructure 90
4.4.1 Inter-die Row Replacement 90
4.4.2 Inter-die Column Replacement 93
4.5 Discussion 94
4.6 Conclusion 97
Bibliography 99

5 **Low Cost Multi-Error Correction for 3D Polyhedral Memories** **101**
5.1 Introduction 102
5.2 ECC for Polyhedral Memories 103
5.2.1 On Die Internal ECC 103
5.2.2 Dedicated Die ECC 104
5.3 Performance Evaluation 108
5.3.1 Error Correction Capability 108
5.3.2 On Die Internal ECC Overhead 111
5.3.3 Dedicated Die External ECC Analysis 113
5.4 Conclusion 115
Bibliography 117

6 **LDPC-Based Adaptive Multi-Error Correction for 3D Memories** **119**
6.1 Introduction 120
6.2 LDPC-based Error Correction Proposal 121
6.3 LDPC Decoder Design Space 125
6.4 Evaluation 128
6.5 Conclusion 132
Bibliography 133

7 **Conclusions and Future Work** **135**
7.1 Summary 135
7.2 Future Research Directions 139

List of Publications **141**

Samenvatting **143**

Propositions **145**

Curriculum Vitae **147**

List of Tables

2.1 TSV Access Control Signals 57

3.1 CMOS vs. hybrid NEMFET-CMOS Caches Comparison. 73

3.2 Processor Configuration. 75

4.1 TSV requirements for proposed schemes 95

4.2 Required TSV pitch [μm] for 16nm 6T-SRAM cell 97

6.1 LDPC Encoders and Decoders in ASIC Implementations 130

List of Figures

1.1 Brief memory technology and computing system evolution road map. 2

1.2 The processor - memory performance gap [8]. The memory baseline is 64 KB DRAM from 1980. In the last decades processor manufacturers gradually introduced several cache levels in an attempt to reduce the gap. 9

1.3 Technology process generations (adapted from [57]). 10

1.4 Microprocessors trend data (adapted from [64]). 12

1.5 State of the art memory hierarchy overview. 13

1.6 3D technology classification. 20

1.7 TSV based 3D integration. 22

2.1 Memory hierarchy. 53

2.2 Traditional cache and memory array layout. 54

2.3 Polyhedral memory array. 55

2.4 Required TSV access logic. 56

2.5 Access time comparison. 59

2.6 Footprint comparison. 60

2.7 Read energy comparison. 60

2.8 Leakage comparison. 60

3.1 NEMFET's geometry [8]. 68

3.2 NEMFET's inverter schematic and transient behavior. 69

3.3 Hybrid NEMFET-CMOS memory cell. 70

3.4 Two adjacent hybrid NEMFET-CMOS memory cells layout. 71

3.5 L1 instruction cache average miss latencies. 76

3.6 L1 data cache average miss latencies. 76

3.7 IPC reduction relative to **baseline** (lower is better). 78

3.8 Energy difference relative to **baseline** (higher is better). 78

4.1 Memory redundancy with external (a) and internal (b) spares. 86

4.2 General 2D internal redundancy. 87

4.3 3D inter-die memory repair - general idea. 88

4.4 Memory partitioning. 89

4.5 Inter-die row replacement infrastructure with idle *provider*. 91

4.6 Inter-die row replacement infrastructure parallel *consumer* and *provider* access. 92

4.7 Inter-die row replacement with “busy *provider* identical access pattern“, divided wordlines, detail. 93

4.8 Inter-die row column replacement infrastructure. 94

4.9 TSV area overhead vs. row area for 32-bit data I/O. 96

5.1 *On die internal ECC* for polyhedral memories. 104

5.2 MEC-based ECC for polyhedral memories. 105

5.3 Word error rate vs bit flip probability. 109

5.4 On die internal ECC evaluation. 110

5.5 On die internal trade offs. 111

5.6 Dedicated die ECC - footprint comparison. 112

5.7 WER vs. ideal scrubbing time. 114

5.8 WER vs. ideal scrubbing energy. 115

6.1 LDPC-based error correction for 3D memories. 122

6.2 Memory scrubbing timeline. 123

6.3 Memory maintenance scrubbing flow. 124

6.4 LDPC decoders summary. 127

6.5 LDPC vs extended Hamming. 129

6.6 Area vs. clock period, area vs. power, area vs. decoding latency, and area vs. decoding energy. 130

6.7 LDPC error correction die real estate utilization. 131

6.8 L2 correction capability. 132

1
Introduction

Memory represents an essential constituent of any computing machine, along with data processing cores and transport infrastructure. It is utilized to store both the to-be-executed program instructions as well as the input/output information data and any other intermediate values. Memory capacity and behavior represent a key factor for the overall system performance and in the following we briefly describe its metamorphosis tightly coupled with the one of computing technology since its infancy up to state of the art, by following the time line depicted in Figure 1.1.

Early Days

In the 1830s Charles Babbage originated the programmable computer concept while conceiving a machine capable of automatically perform a large variety of mathematical calculations. He named it the *analytical engine* and his intentions were to revolutionize computation by mechanizing it. He saw this as highly necessary since the computer term in his time was referring to actual humans that were performing mathematical calculations in a very time inefficient and error prone manner, relying on mathematical tables printed on thousands of pages [1]. When Babbage was designing the analytical engine Faraday's work on electromagnetism was just about getting under way, while electrical circuits were only mathematically analyzed by Ohm. Hence, Babbage had to rely on existing technology, which at that time could provide mechanical components [1], with the steam engine being the only valid option to drive the entire assembly.

If built, the analytical engine would have weighted several tones, with the store unit, i.e., the today modern computer main memory equivalent, measuring 7 m in length and 3 m in height and width. It would have operated on 50 decimal

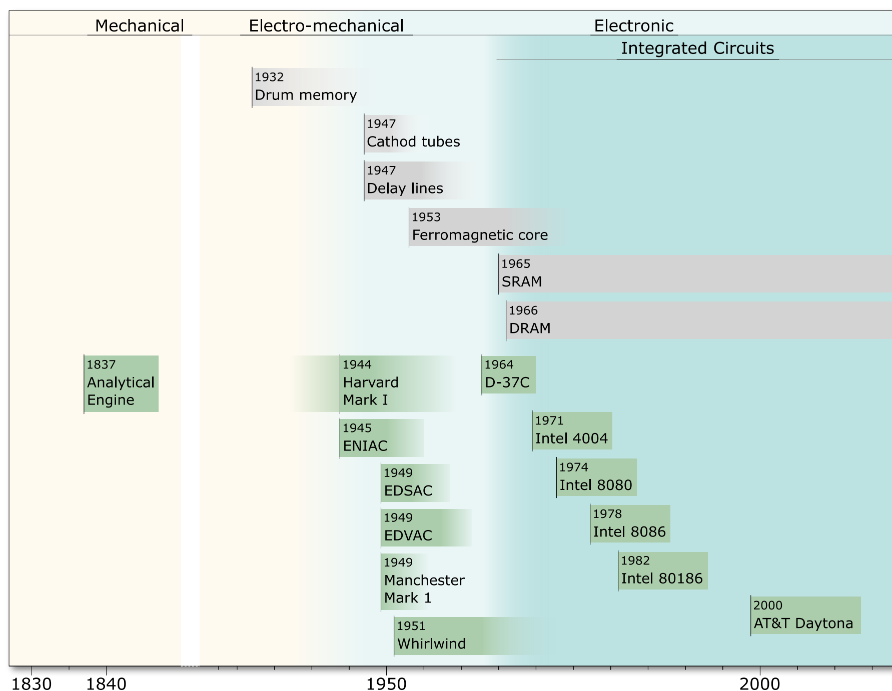


Figure 1.1: Brief memory technology and computing system evolution road map.

digit numbers, with the store unit consisting of about 1000 such entries. For each digit, i.e., a basic memory element, a 10-toothed metal wheel would have been utilized. An elaborate mechanical transportation way based on racks and pinions would fetch the decimal stored numbers to the processing unit, entitled by Babbage as the mill, since, once the numbers arrived at the input, they were milled around to get the results. Reliable Jacquard punched card loom technology would have been used for program input [2]. Even though Babbage was unable to construct the analytical engine, it is considered one of the first general purpose computer, since it was designed to contain all the modern computer core components [2]. A key element in the analytical engine design was the ability to perform conditional branching. This, together with a potentially theoretical limitless memory store unit, makes the analytical engine a Turing complete machine, a concept unheard of at that time.

Inspired by Charles Babbage's work, Howard Aiken designed in the 1930s a general purpose computer, named Mark I [3]. In contrast to his predecessor, Aiken was living in an era in which electronic components developed substantially to allow for their utilisation in computers. Mark I was an electromechan-

ical computer and was constructed during the 1940s by IBM in collaboration with Harvard University. Its memory consisted of 72 adding storage and 60 constant registers, each 23 decimal digits long, implemented as electromechanical counter wheels. Punched paper tapes together with switch positions were employed for program input [4].

Shortly after Mark I's development, which was to remain operational until 1959, electromechanical computers became obsolete as they were too slow paced for the highly dynamic post WWII world. The first fully operational electronic computer, called ENIAC (Electronic Numerical Integrator and Calculator), was built in the mid 1940s [5]. Even though it was designed for a specific military goal, i.e., to compute artillery firing tables, it was assembled too late to be used in the Second World War effort. Its first task still had a military purpose, being utilized to perform a series of complex calculations related to hydrogen bomb development. However, it was later utilized as a general purpose computer solving a steady stream of problems in a variety of fields ranging from number theory to meteorology, that demonstrated its versatility and power [2] and it remained operational until 1959 when it was disassembled. ENIAC was also a decimal machine, as its (electro)-mechanical predecessors. Its memory consisted of 20 accumulator registers, each capable of holding a 10-digit decimal number, with a ring, the electronic counterpart to the mechanical calculator's decimal wheel [2], of 10 vacuum tubes used for each digit [6; 7].

ENIAC's major drawback was that programming was manually executed by plugging up cables and setting switches, with input data being provided on punched paper cards. Reprogramming required from a half an hour to a whole day [7; 8]. The team behind the ENIAC, including John von Neumann, was aware of its time pressure induced limitations which resulted in a difficult machine programming manner, and wanted to improve this aspect. Already in 1945, when ENIAC was not yet entirely finalized, they were discussing about storing programs as numbers and published a first report draft [9] proposing a stored-program computer called EDVAC (Electronic Discrete Variable Automatic Computer), the successor of ENIAC. This was an inspiring memo that originated the stored-program concept, which was right away adopted by computer researchers worldwide. The stored-program concept is still the basis for nearly all digital computers today [7; 8]. Maurice Wilkes and his team from Cambridge University employed it during the construction of what is considered the world's first full scale operational stored-program computer [8], the EDSAC (Electronic Delay Storage Automatic Calculator), which became operational in 1949. The EDSAC logic was implemented by vacuum tubes and its

memory by ultrasonic mercury delay lines, a technology with origins in radar. The total memory consisted of 1024 storage locations, each being capable of holding a number with 16 binary digits plus a sign digit [10; 11]. In delay line memories, trains of electronic pulses are converted by means of quartz crystal into ultrasonic pulses, which are inserted into a special medium path that slows them down [12; 13]. From the delay element end they are fed back to the beginning through amplifying and timing circuits. Thus, delay line memories required periodic refresh in order to maintain their stored values, and provided variable latency sequential access only, for which complex extra access logic was required. However, they were much more reliable and cheaper then vacuum tubes flip-flops memory instances utilized, e.g., in ENIAC.

For sequential access memories data are always accessed one after the other, which allows for quick retrieval times when program execution does not encounter any branches, or when the operands reside one after the other. However, if an arbitrary address is encountered, the requested location needs to be sought first, an operation that can be extremely time inefficient. This was the case for delay line memories for which the seeking operation incurred unexpected delays and by implication program behavior. Thus, there was a stringent need in those times for the processor to be able to access a memory location in almost the same time interval, irrespective of the data physical location inside the memory, in order to ensure reliable execution times estimation. One of the first such Random Access Memory (RAM) implementations was the Williams-Kilburn tube, which stored data as electrically charged spots on the face of a cathode ray tube [14]. It was firstly utilized in the Manchester Small-Scale Experimental Machine (SSEM), which was not designed to be a practical computer, but to act as a proof of concept for the Manchester Mark 1 machine [15; 16]. The Manchester Mark 1 had two Williams-Kilburn CRTs as main memory holding a total of 128 words each 40 bit long. The today's equivalent of disk storage consisted in about 3000 words and was implemented by means of drum memory, a type of memory that stores information on the outside of a rotating cylinder coated with ferromagnetic material, which is circled by read/write heads in fixed positions [17]. Manchester Mark 1 developed into Ferranti Mark 1, which became the world's first commercially available general-purpose electronic computer in 1951 [18].

The Massachusetts Institute of Technology Servomechanisms Laboratory developed for the U.S. Navy between 1948-1951 the Whirlwind I computer [19]. Williams-Kilburn tube RAMs were among the considered options for the Whirlwind I memory implementation. However, to avoid the Williams-Kilburn tube required refresh times, a special kind of CRT electrostatic storage

tubes were chosen instead [20; 21], a decision that soon proved to be costly. The team behind the Whirlwind project quickly renounced the unreliable electrostatic storage tubes in favor of the newly proposed ferromagnetic core memories [22] for Whirlwind II [23], which was the first computer to make use of such a technology for its 2048 16-bit word memory. This proved a wise decision since magnetic core memories, which consisted of an array of multiple ferrite rings that exhibit a hysteresis curve, were by far more reliable than CRTs (or mercury delay lines), much cheaper, and required considerable less real estate. They soon became the element of choice for main memory implementations during the 1950s and were heavily utilized until the 1970s [24; 25].

By the beginning of 1950s computing systems were employing a hierarchical memory system: core memories were backed up by magnetic drums which were, in their turn, backed up by magnetic tapes. Starting from late 1950s a fast but small magnetic core memory was added as a slave for a slow, but much larger one, in such a way that the effective access time is nearer to that of the fast memory than to that of the slower memory. This marked the beginning of caches, which were formalized as a concept by Wilkes in 1965 [26]. The programmers dream of having almost unlimited amounts of fast memory at their disposal seemed somewhat possible due to the incipient memory hierarchies that ferromagnetic memories facilitated. This was however more achievable with the next significant technological discovery, i.e., the transistor, that became the fundamental building block of modern electronic systems. The field effect transistor was proposed as a concept in Canada by Julius Edgar Lilienfeld in 1926 [27] and in Europe by Oskar Heil in 1934 [28] in a time when it was not possible to actually construct a working device, and no one was able to do anything with it [29]. The breakthrough came in 1947 when John Bardeen, Walter Brattain, and William Shockley from Bell Telephone Laboratories implemented the first working bipolar transistor device [30]. Transistors quickly replaced electronic vacuum tubes in similar circuit configurations since they offered tremendous power reductions and smaller form factor, which resulted in considerable speed improvements and, as a side note, reduced for the moment the need for air conditioning [24]. Transistor logic, together with core memories brought significant improvement in reliability and there were no longer talks about ”mean time to failure” [24], and remained essential computer building blocks for several years. The Manchester University experimental transistor computer, to become operational in 1953, is considered to be the first transistor based computer implementation.

Integrated Circuits Revolution

Transistors contributed in a substantial manner to computer development, but very soon manufacturers were confronted with a problem that seemed insurmountable: the huge component number involved in projects was almost unmanageable in terms of design complexity and necessitated an excruciating and error prone effort in terms of hand wiring. For several years researchers went through what Bell Labs vice president Jack Morton entitled as "the tyranny of numbers" [31]. The solution came in 1959 when Jean Hoerni and Robert Noyce, both from Fairchild Semiconductor, introduced the planar manufacturing process [32] and the monolithic approach [33], respectively. These proposals completely revolutionized the semiconductor manufacturing technology and the Integrated Circuits (ICs) era started. The first working monolithic ICs were produced in 1960, credited to Jack Kilby from Texas Instruments [34] and Robert Noyce from Fairchild Semiconductor [33]. ICs from the early 1960s were manufactured with a low density fabrication process, known as Small Scale Integration (SSI), consisting of only a few transistors and provided limited computation functionality. Military and aerospace computer systems were among the first to benefit from the ICs applications in the mid 1960s, e.g., the D-37C computer used in the LGM-30 Minuteman intercontinental ballistic missile project in 1964 and the Apollo Guidance Computer developed by NASA in 1966 [35]. The integration scale rapidly rose though, approximately doubling every year, and by the end of 1960s Medium Scale Integration (MSI) emerged, with a chip transistor count of 100 or more. As ICs complexity increased, practically all computers switched to IC-based designs since more functionally complex logic devices such as adders and shifters could be easily implemented on a chip [36].

While processors were already benefiting from IC emergence, memories were still implemented based on ferromagnetic core technology, but this was about to change soon. In 1961 Bob Norman from Fairchild Semiconductor thought that it was practical to begin considering semiconductor devices implementing a flip-flop array for memory storage [37] and four years later the first Static RAM (SRAM) was proposed by Scientific Data Systems and Signetics. Implemented using bipolar technology it had 256 72-bit word capacity and an access time of 120 ns [38; 39]. Two years later the single transistor Dynamic RAM (DRAM) was invented by Robert Dennard from IBM [40; 41; 42], and in 1970, Intel, at the time a small start-up company specialized in memories [43], produced the first commercially available DRAM. The Intel 1103 MOS DRAM had a capacity of 1-Kb, an access time of 300 ns, and was nicknamed

”The Core Killer” [39]. This was the beginning of the end for ferromagnetic memories, which by the mid-to-late 1970s were completely replaced by semiconductor memories as the primary computing systems storage devices.

Intel was part of another important evolutionary step in 1971 with the introduction of the first single-chip microprocessor, the Intel 4004 [44; 45]. Produced in the beginning of the Large Scale Integration (LSI) period, Intel 4004 processor was fabricated with a 10 μm silicon-gate enhancement load pMOS technology, consisted of 2300 transistors, and was able to operate at a maximum clock frequency of 740 KHz [45; 46]. Implemented on a 12 mm^2 die, this revolutionary microprocessor could execute a total of 0.092 Million Instructions Per Second (MIPS), which was almost twice as much as the first electronic computer, the ENIAC, which filled an entire room [47]. The 4004 microprocessor was part of the Micro Computer System 4 (MCS-4) chip-set that included three other chips, namely 4001 Read Only Memory (ROM), 4002 Random Access Memory (RAM), and 4003 Shift Register [46], and allowed for the easy realization of small computers with varying memory amounts and I/O facilities. Subsequently, in 1974 Intel introduced the 8080 microprocessor that operated at a maximum clock frequency of 3 MHz [48]. This is considered the first true general-purpose microprocessor [49], since it did not require the presence of additional chips as it was the case for 4004, and it was the core of Altair - the first personal computer.

Since 1970s ICs density underwent a staggering revolution following quite rigorously the insightful prediction initially stated in 1965 [50] and further formalized in 1975 [50] by Gordon Moore, that per die transistors count will grow exponentially. Moore’s initial projection was for at least a decade, but being adopted as a target guideline by the semiconductor companies in order to maintain their market position, it prevailed for the past 5 decades. From 2300 transistors, for Intel 4004 in 1971, using the 10 μm process technology, to 7200000000 transistors, for Intel Xeon Broadwell-E5 in 2016, using 14 nm process technology, is a six order of magnitude increase. This technology development allowed for different capabilities to be gradually included for processor performance improvements. For example, in the beginning of 1970s the transistor high rate count increase was used to advance the architecture from 4-bit, to 8-bit, to 16-bit, and to 32-bit microprocessors in the early 1980s, when between 25000 and 50000 transistors could fit on a single chip [8]. 32-bit processors remained the norm until the early 1990s when they were slowly replaced by their 64-bit successors, which are still manufactured today, while 128-bit instances not being expected too soon.

The above presented technology evolution enabled for a $10^8\times$ MIPS increase over the last six decades [51], while the computing power cost has decreased exponentially [52]. Many challenges were overcame in order to maintain a sustained miniaturization trend that allowed for today's pocket-sized devices to be more powerful than 1960s-1980s supercomputers that took up entire rooms. Subsequently, computing systems were uniquely adopted and spread virtually to all economy sectors, from defense industry, aerospace, telecommunications, to automotive, health, and food industries. However, in order to keep up with the continuously developing market requirements, many challenges are still to be faced by computer designers, with the most important ones being present in the next section while keeping the main focus on memory hierarchy technology and architecture.

1.1 Modern Processor Design Challenges

Initially, microprocessor chips were operating at smaller frequencies than main memory DRAM chips, together with which they were teamed up to form computing systems since 1970s, while by the beginning of the 1980s, microprocessors and memories reached somewhat similar operating frequency levels. The Intel 8086 processor, launched in 1978, which would give rise to Intel's most successful line of processors, the x86 architecture family, operated at 5 MHz frequency, i.e., 200 ns clock period, while its successor, Intel 80186, launched in 1982, operated at 6 MHz, i.e., 166 ns clock period. At the same time, in 1980, a 64 Kbit DRAM had a row access time between 150-180 ns and a (read/write) access time of about 250 ns[8]. This balance abruptly changed by the end of the 1980s when logic performance started to substantially surpass memory performance and the well known processor-memory bottleneck [53] (also known as the memory wall [54]) emerged. As detailed in Figure 1.2, microprocessor performance increased $1.52\times$ per year between 1986 and 2000, and $1.2\times$ per year between 2000 and 2005, while DRAMs experienced only a $1.07\times$ yearly performance increase over the 1980-2005 period [8].

While logic operations once represented the processor bottleneck, the 1980s fabrication technology revolution (scaling) allowed for, e.g., faster clock rates, extensive instruction level parallelism with about 20 stages deep pipelines, out-of-order execution, and sophisticated instruction fetch units. These developments, combined with the aforementioned gently DRAM performance increase, led to a data availability crisis. Considering a hypothetical and greatly simplified scenario with a processor operating at 800 MHz and a memory run-

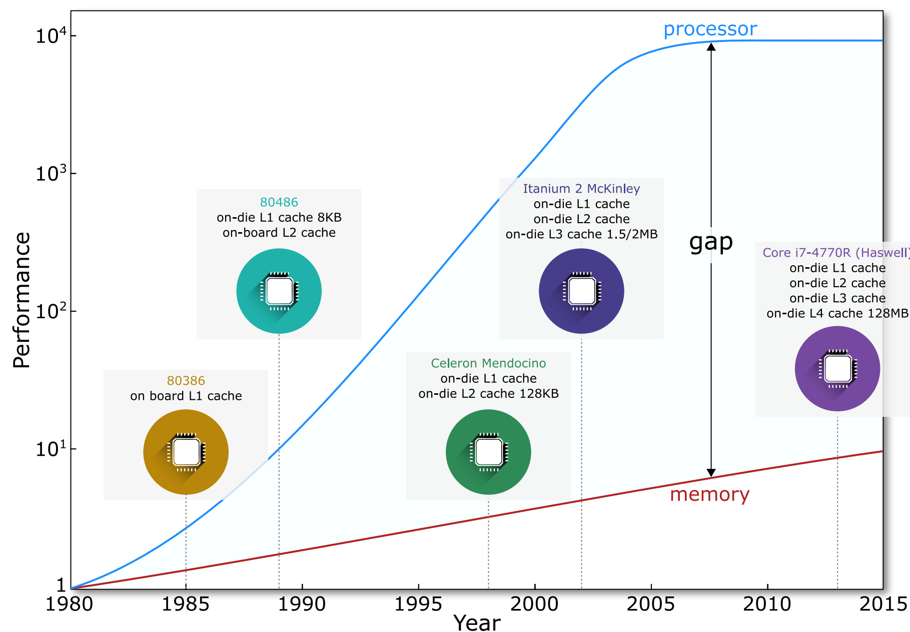


Figure 1.2: The processor - memory performance gap [8]. The memory baseline is 64 KB DRAM from 1980. In the last decades processor manufacturers gradually introduced several cache levels in an attempt to reduce the gap.

ning at 100 MHz, for each single memory access, 8 processor clock cycles are elapsed. This could lead to the processor being forced to wait 7 cycles until the data that reside in the memory are retrieved, an inefficient situation since those 7 cycles are practically waisted.

In an attempt to bridge the processor - memory gap, i.e., to be able to feed the processor with enough data to diminish the number of memory waiting generated stall cycles, different solutions were employed. Wilkes' idea of caching, proposed for core memories in the 1960s, was among the considered approaches and it proved to be somewhat successful. To benefit the most from this solution a faster memory than DRAM was required. SRAM proved to be the best cache implementation choice, since, even though much more expensive than DRAM, it was much faster. Initially, caches were introduced on-board, e.g., for Intel 80386 launched in 1985 [55; 56], and then on-chip, when the semiconductor technology reached a certain point and was able to eliminate the chip crossings between processor and cache memories by their co-integration. The first processor with an integrated on-chip cache was Intel 80486 introduced in 1989. To further mitigate memory speed and capacity re-

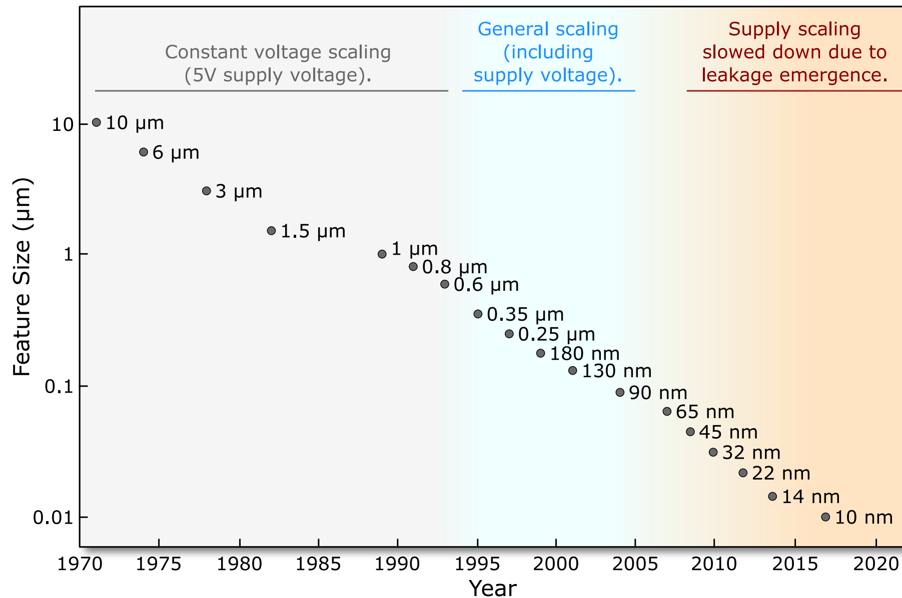


Figure 1.3: Technology process generations (adapted from [57]).

quirements, after almost a decade a second cache level was added in 1998 for the budget oriented Celeron Mendocino processor. Very soon after, in 2002, the third cache level was introduced for the Itanium 2 McKinley processor, that was targeted towards enterprise servers and high-performance computing systems. Other manufactures followed the same scenario and at present times a three level SRAM cache hierarchy is the norm. Quite recently, in 2013, Intel included a fourth cache level implemented with embedded DRAM (eDRAM) technology. The complex multi-level cache hierarchy present in modern processors was enhanced by the aggressive technology scaling which allowed for increased integration density, a topic that is further detailed in the next section.

1.1.1 Technology Scaling

The feature size of a CMOS process refers to the minimum transistor length with which devices can be reliable manufactured [57]. Continuous technology scaling consisting in a feature size reduction by 30% every two to three years, as depicted in Figure 1.3, was a major driving force behind constant improvements in ICs performance, area and power reduction. At the core of this process was Robert Dennard's observation from 1970s that the basic transistor operational characteristics can be preserved and its performance improved

when some critical parameters are scaled by a dimensionless factor [58]. These parameters include device dimensions (i.e., gate length, width, and oxide thickness) and voltage, that are downscaled, and doping concentration densities, that are upscaled by the inverse of the same factor.

The reduction of the power supply voltage was considered by the industry undesirable from a system level perspective since it would have break the plug-in compatibility with every new generation [59]. Hence, a 5 V constant supply voltage scaling was performed up to through the early 1990s, an approach that offered quadratic delay improvement and maintained continuity in I/O voltage standards, but it increased device breakdown risk and power consumption become unacceptable [57]. Therefore, since the half-micron node downward general Dennard scaling became the norm, even though the supply voltage did not follow exactly the scaling factor applied to device dimensions [57], and was very successfully employed by the industry up until the 130 nm generation node in the early 2000s. At that technology node the gate oxide thickness reached about 1.2 nm and electron tunneling through such a thin dielectric lead to an abrupt increase of the leakage current. This posed significant challenges to technology scaling that were further addressed by introducing innovations in transistor materials and structure [60]. Examples in this direction include: strained silicon transistors introduced by Intel for the 90 nm technology [61], high-k metal gate transistors utilized in Intel's 45 nm technology [62], and the more notorious FinFET (tri-gate) transistors employed in Intel's 22 nm technology [63].

Technology feature size shrinking was a major driving force behind ICs transistor count exponential growth and, since with scaling transistors generally improve in performance, behind the continuous clock frequency increase, which, e.g., for Intel's microprocessors, doubled approximately every 3 years from their introduction until 2005. This contributed to a sustained overall processor performance improvement, as summarized in Figure 1.4, that, even though transistor scaling continued, began to saturate by mid 2000s. The main reason behind was that, in contrast to transistors, IC wire delays do not diminish with decreased feature size. This may not seem to be a problem for local wires, that run within functional units and which are short enough for their resistance to not matter, or for semi-global wires, that run between functional units and for which an appropriate number of repeaters could result in enough speeding up [57]. For global wires however, that run across an entire chip, even with multiple repeaters their delays are getting bigger as technology scales. Thus, the time to cross a chip in a nanometer process increases substantially, making wire delays becoming the dominant issue for each clock cycle

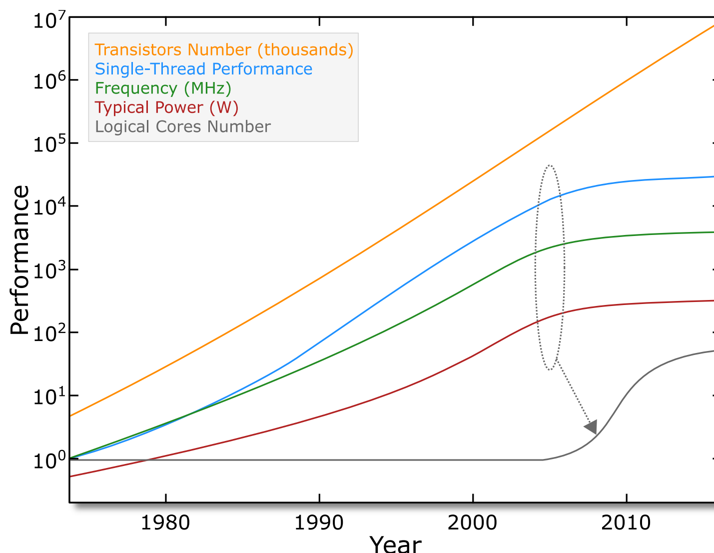


Figure 1.4: Microprocessors trend data (adapted from [64]).

and a major design limitation for large integrated circuits [8; 57]. In addition to wire delays, power developed also into a real concern for circuit designers and the combination of the two led to a stall in uniprocessor frequency increase and has limited the instruction level parallelism exploitation. Aware of this concerns and since it was easy to envision that the integration level was about to soon allow for a billion transistors per chip, in 1990s several studies suggested that both software and technology trends have reached the point when one should start favoring chip multiprocessor microarchitectures [65].

One of the first manufactured multiprocessor chip was AT&T Daytona [66], a four core digital signal-processor, introduced in 2000 [67]. However, the real paradigm switch occurred only after 2005 when multiple general purpose multi-core processors reached the market [8; 67]. Since then Intel has been adding about two more cores per generation and performance increase became more of a programmers burden in that it requires a switch to a parallel software design paradigm [8].

After the 2005 shift towards multi-core chip processors there has been no substantial improvement noticed for frequency and single-thread performance. The focus moved on memory hierarchy, to improve data availability, and other architectural concepts related to parallel computing. To get some insight into those developments we depict in Figure 1.5 a state of the art multi-core processor memory hierarchy. At the top the register files are the smallest but fastest

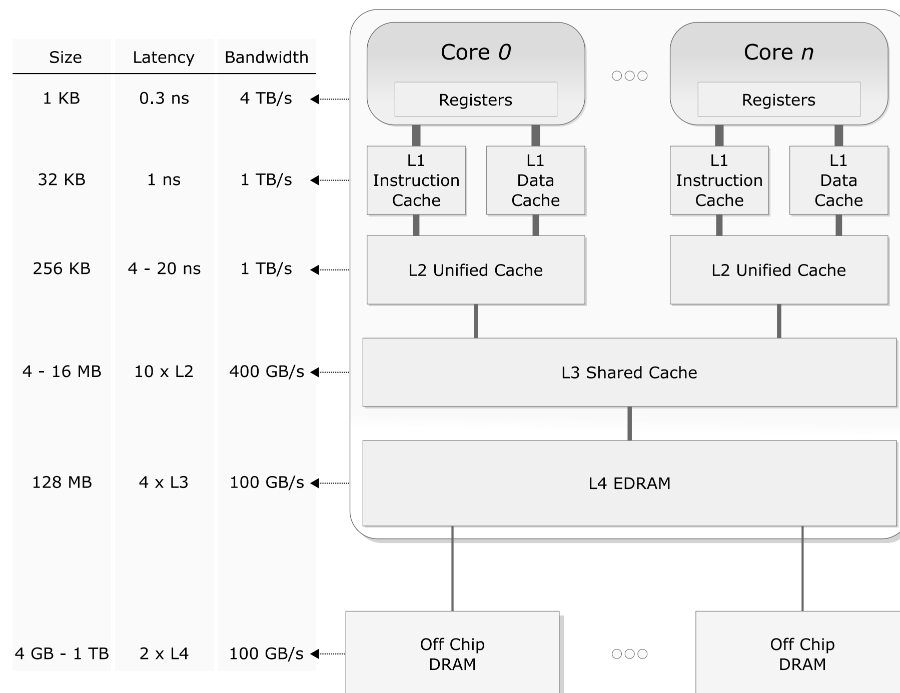


Figure 1.5: State of the art memory hierarchy overview.

memories that operate at the core frequency, in the order of a few GHz, and which are supported by separate instruction and data L1 caches, each of about 32 KB and operating at about 1 GHz frequency. A unified L2 cache level follows, which is about 8× larger than an L1 instance, 4 to 20× slower, and feeds both L1 caches. The next cache level is shared by all the cores, being slightly larger than the L2, and about 10× slower. This shared L3 can be optionally followed by an 128 MB 4× slower fourth level cache implemented with eDRAM technology, which constitutes the last on-chip memory element. Further down the hierarchy, on the motherboard, several memory channels ensure communication with external DRAM chips, which are 2× slower than the L4, but offer several times more storage capacity.

Having presented the modern processors' state-of-the-art in terms of technology and architecture, we next focus on detailing the challenges currently faced by computing system designers.

1.1.2 Ensuring Data Availability in Multi-core Processors

It can be noticed from Figure 1.5 that multi-core processors put more pressure on the memory hierarchy since they significantly increase the bandwidth requirements. This is easily observable for the shared L3 cache, since in the multi-core context it has to supply data for multiple L2 caches instead of only for one, as it used to be the case for single-core processors. As a result, L3 generates more requests further down the memory hierarchy leading to a natural off-chip DRAM traffic escalation when the cores number increase. With a memory bandwidth increase limited by the pin count growth, which is about 10% per year, and considering that the an on-chip core number is doubling every 18 months, today's processor designers are facing a bandwidth wall [68]. The provided off-chip memory bandwidth should sustain the rate at which memory requests are generated, otherwise adding more cores to the chip will not yield any additional performance. Thus, it is futile to expect overall performance gains from multi-core systems proportional to the increase in the cores number without substantial memory and its afferent interconnection redesign.

Several methods to increase the off-chip bandwidth have been proposed. A shared DRAM controller design that provides a fair thread DRAM access policy to boost system throughput is introduced in [69]. A self-optimizing memory controller design that operates using the principles of reinforcement learning is presented in [70]. In [71] the authors propose to exploit processor inactivity periods due to memory stalls, i.e., when the processor is waiting for data to be fetched from the off-chip memories, in order to improve off-chip bandwidth. Based on the fact that in those pausing intervals the processor requires much less power, since the frequency is usually scaled down some power delivery pins become available and can be used for data transport to ensure extra bandwidth. Unfortunately, while being able to bring certain benefits, the presented methods off-chip bandwidth increase is limited, since in the best case it can only reach a theoretical maximum, which considering the current pin number constraints, is not sufficient to feed the multi-processor cores with enough data.

A step forward from the technology perspective is represented by the emergence of three dimensional stacked ICs, which seem to provide a promising solution to this DRAM bandwidth problem, since they allow for an amount of interconnection unconceivable in current 2D off-chip designs [72]. While we detail this approach later in this chapter (see Section 1.2), it is important to note here that it puts additional pressure on the last level shared cache, which is expected to become the data traffic bottleneck at the processor to DRAM

frontier. Thus, it is of high interest to investigate novel multi-port memory designs to be employed as shared caches, such that a sustained high amount of parallel accesses can be served and processor data availability is ensured, a topic that is addressed in Chapter 2 of this thesis.

In the next subsection we focus on static power dissipation, which reduction represents another modern processor design challenge that emerged due to technology scaling.

1.1.3 Processor Design Within a Restricted Power Envelope

CMOS technology was renowned for its manufacturing process simplicity and its almost ”zero” static power consumption characteristic, aspects which contributed substantially to its dominance. However, already since reaching the Very-Large-Scale Integration (VLSI) level, that enabled ICs to hold hundreds of thousands of transistors, power consumption developed as the most important constraint in circuit design, overcoming traditional area and delay metrics [73]. Generally speaking, CMOS circuit power dissipation can be broken up into three main categories: dynamic, short circuit, and static power. In the past, its dynamic constituent represented the main dissipation source and performance-driven technology scaling allowed for substantial dynamic power consumption reductions. However, since the MOSFET threshold voltage scalability frontier limits the power supply voltage reduction, technology scaling is no longer able to sustain the same amount of power reduction [74]. In addition, static power, which is dissipated by leakage currents that flow even when the device is inactive, increases abruptly with technology scaling and becomes a significant component of the overall power consumption [75; 76]. There are several leakage sources for MOS transistors [77], with the most significant being considered the subthreshold leakage, the gate leakage, and the junction leakage [57]. According to ITRS [78] subthreshold and gate leakage are projected to exponentially increase with MOSFET feature size scaling. The gate leakage current is relatively smaller then the subthreshold leakage for older technologies, but it is expected to be comparable to the subthreshold leakage current in the deep nanometer nodes [77].

Since in modern chip multiprocessor systems a significant real estate is dedicated for storage, in some cases considerably larger than 50%, memory represents a major contributor to the overall system power consumption [8; 79; 80; 81; 82; 83; 84]. Experimental results from [80] indicate that the leakage power consumption of L2 caches becomes almost 40% of the total power of cores, interconnect, and L2 caches. Simulations from [84] suggest

that the total static energy ranges from 80%, for a two-core two-issue 8 MB L2 processor, down to 30%, for an eight-core eight-issue 1 MB L2 processor. For mobile targeted devices, in which the processors are designed in a less aggressive manner in order to fit within a tight power budget, the caches can account for 25% to 50% of the total power consumption [8]. Thus, reducing the memory power consumption directly implies an important overall energy benefit for chip multiprocessor systems and there has been a large body of work in this direction.

Several circuit level memory power reduction techniques were proposed, e.g., forced transistor stacking [85] or PMOS-based pull-up networks with domino logic [86]. Alternatives that combine circuit with architectural techniques were also introduced. One of the first such approach [87; 88], which is also mostly employed, consists in gating (shutting off) the SRAM cells comprising inactive cache blocks. The decision upon which cache blocks are turned off is crucial since it is highly desirable to obtain significant power benefits with little penalties in terms of performance. For single threaded single processor systems the decision can be taken by considering the characteristics of the only executing program, thus cache blocks are turned off if they are not accessed for a predefined, or dynamically adapted, threshold time-out cycles [89; 90; 91; 92]. For chip multiprocessor systems many challenges are present since multiple threads are simultaneously executed on multiple processors and their cache blocks might interact with each other. Several methods were proposed, including virtual exclusion [93], an architectural technique that saves L2 caches leakage by keeping the data portion of repetitive but infrequently accessed cache lines off, given that locality allows for such entries to be already present in L1. Other approaches consist in turning off cache lines by using the coherence protocol invalidations, utilizing cache decay techniques specific for coherent caches [94], employing a dynamical way adaptation mechanism [95; 96], or by exploiting replicated cache blocks [80]. It is important to mention that for all the above techniques performance penalties are incurred.

Despite of all the above efforts, a static power increase with threshold voltage reduction is always expected for future CMOS devices. This represents a serious technology scaling limitation that generates a stringent need to investigate alternative memory designs based on the utilization of emerging devices with ”zero” leakage currents. Before completely moving to novel emerging devices, considering the CMOS technology maturity, hybrid approaches seem to be the next natural step forward, a direction which is further explored in more detail in Chapter 3 of this thesis.

In the next subsection we focus on the last modern processor design challenge addressed in this dissertation, the ensuring of dependable memory operation in the context of increased technology scaling induced device variability.

1.1.4 Memory Dependability in the Nano Scale Era

In addition to the above presented static power wall emergence, with technology feature size reduction increasingly smaller geometries should be handled. This leads to a less precise manufacturing process, with more defects being induced, which make transistors more prone to various in-field failures. Maintaining ICs reliability at the desired market demanded level became a critical challenge which is to be addressed both at design-time and at runtime [97].

Memories are susceptible to both hard failures, related to the physical breakdown of a device or an interconnect, and soft errors, usually triggered by radiation striking a chip that corrupts the stored information. Considering that memory cells are typically designed with a minimum silicon area and occupy most chip real estate, memories are most sensitive to process variations and device mismatch than other semiconductor-based electronic devices. Fortunately, their regular structure makes it easy to enhance their design such that repair mechanism are incorporated to improve reliability. Such a method consists in the addition of redundant spare rows and columns for hard error correction. This dates back many decades, being firstly proposed and utilized for 64 Kb DRAMs in 1979 [98]. The employed spare types as well as their number depend on the anticipated defect density, with the important mention that even a small amount of redundant elements can make a big difference in yield. Originally, testing and repair, i.e., bad rows/columns disabling and spares allocation, were performed in factory, with special equipment being required for that. With device miniaturization, it became possible to incorporate built-in self test and repair mechanisms in memory chips and allow for on-field test and repair.

Since scaling trends have considerably reduced memory cell size, an increasing soft error rate has been observed. This is due to the smaller charge stored on the memory cells and the considerable increase in chip storage capacity [57]. Memories became extremely sensitive to soft errors and Error Correction Codes (ECCs) have been utilized to cope with this issue. Extended Hamming ECCs [99], which are still the norm in this respect, use additional 8 check bits to any 64-bit memory word to correct one error and detect any pair of errors in the word. The additional check bits and the corresponding mechanism increase memory area and add delay penalties, but Single Error Correction

Double Error Detection (SECDED) ECCs able to make use of information redundancy to improve reliability have been utilised for decades. However, once entering the 100 nm era, Multi-Bit Upsets (MBU) become much more frequent since a particle hitting a memory array creates disturbances in multiple physically adjacent cells [100; 101; 102; 103]. Column interleaving methods [104; 105; 106] were proposed as a solution but this seems not to suffice any longer as according to [107] an MBU multiplicity of over 100 bits is predicted for 32 nm and 22 nm SRAM generations. This clearly suggests that state of the art schemes are not effective any longer and research effort is required towards improved ECC techniques capable to cope with the increasing error amount.

In view of the above, it is crucial to investigate alternative avenues that enhance memory dependability. More specifically, designs that allow for increased spare access could provide improved possibilities for memory repair. In addition, Multi-Error Correction (MEC) codes might provide the means to mitigate the detrimental effects of high MBU multiplicity. We follow in this thesis both directions in Chapters 4, 5, and 6. The context of the emerging technology enablers is detailed in the following section.

1.2 Technology Enablers

With CMOS technology reaching its scaling limits [108; 109] and the stringent need to maintain the same computing systems performance rate gains, a multitude of alternatives are being investigated. In the following we provide a brief overview of emerging technologies that could continue the same miniaturization and performance trends established in the Moore's law context, while providing means to overcome the processor design challenges detailed in Section 1.1.

1.2.1 3D Integration Technology

Conventional ICs are sandwich-like structures created by superimposing several layers of conducting and insulating materials. The base surface for the manufacturing process comes in the form of a silicon wafer that serves as both mechanical support and electrical common point. Utilizing a fabrication sequence that involves a series of deposition, lithography, etching, and implant steps, a transistors layer is firstly built on the silicon substrate, after which several metal interconnect layers are constructed [57]. By harnessing the third vertical dimension, the technology to manufacture 3D integrated circuits can

vertically stack multiple such traditional IC layers. 3D integration technology is emerging as a promising avenue to overcome some physical, technological, and economical limits encountered in conventional ICs. In the following we detail the motivations behind the 3D technology development, afterwards continue with a brief 3D technology manufacturing classification, and end this subsection by presenting state of the art 3D technology utilization proposals for memory hierarchies.

Motivation

The development of the 3D integrated circuit technology has been driven by the need to provide viable solutions to issues that cannot be any longer properly addressed only within the framework of planar IC designs as follows:

- *Interconnect latency*
  The continuous transistor feature size down-scaling has shifted the dominant latency factor from devices to interconnection wires [110]. Contemporary TSVs are copper pillars with electrical properties comparable with normal metal wires. By placing adjacent blocks on top of each other, millimeter long wires can now be shortened to the micrometer lengths of the vertical interconnections. In [111] it is theoretically proven that by 3D stacking global interconnect wire length scale with the square root of the stacked layer number.

- *Power consumption*
  Shortening of wires also has a direct effect on power usage as less heat is dissipated in the wires. Although a vertical interconnection is expected to dissipate more heat than a normal horizontal metal wire, the significant reduction in total wire length plus the fact that wire repeaters are no longer required, compensate so that the global power consumption is lower for a 3D stack. Even more, larger power savings are obtained through the removal of now obsolete power-hungry transceivers for inter-die communications links, i.e., high speed parallel and/or serial links between processing cores or between cores and memory, in 2D system in package realizations.

- *Heterogeneous integration*
  The ability to integrate heterogeneous technologies onto the same chip has been one of the first and main drivers as there are a large number of applications which can benefit greatly from this, ranging from micro-

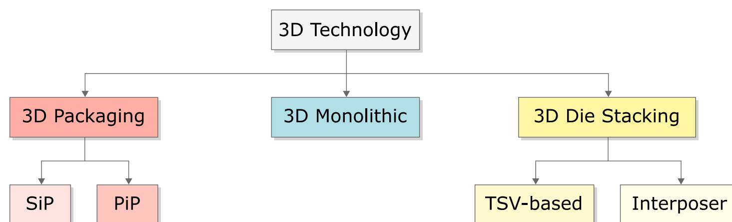


Figure 1.6: 3D technology classification.

and nano-sensors with Micro-/Nano-Electro-Mechanical-Systems and CMOS logic layers to high-performance computing cluster nodes with optical, memory and logic layers. Furthermore, various emerging technologies proposed to replace CMOS-based computing can use 3D technology as an enabler for hybrid transitional circuits.

- *Form factor*
  Apart from system architecture related aspects and performance, the form factor of the complete system can be a key factor precluding the utilization of 2D circuits in space confined applications. By allowing for vertical chip grows 3D stacking opens more miniaturization opportunities of which many applications could benefit from and in this way enable new applications introduction [112].

3D Technology Classification

In the last few years, 3D technology has garnered a lot of interest and significant progress towards commercial availability has been made. Several manufacturing approaches are currently prototyped consisting of three main classes, as depicted in Figure 1.6: 3D packaging, 3D monolithic, and 3D die stacking.

In *3D packaging*, multiple dies are stacked vertically at the packaging level. Interconnects between the dies I/Os are typically formed by wire-bonding, flip-chip, or Ball-Grid-Array (BGA) stacking, approaches that provide the lowest interconnect density from all the mentioned 3D classes. A subclass of the 3D packaging approach is the System-in-Package (SiP), in which the system comprises severals stacked ICs internally connected by fine wires bonded to the substrate and packaged in a single chip. Being the most mature approach, SiPs are widely employed in mobile devices, digital cameras, and portable audio players [113; 114]. Another 3D packaging subclass is Package-On-Package

(PoP) technology where multiple packaged chips are stacked vertically [113]. This can include several SiPs packages.

In the *3D monolithic* approach [115] the first circuit layer is constructed in a similar fashion to traditional 2D CMOS ICs. Next, a dielectric is added on what would have been the last metal layer for the 2D only approach, on which a new circuit layer consisting of transistors and several metal layers is built. This procedure is repeated until the desired number of 3D circuit layers is achieved, which are interconnected using nano-scale inter layer vias, with no bonding materials being required between the circuit layers. While the monolithic approach provides the highest vertical interconnect density between the stacked layers, it is very challenging to ensure a low-temperature process for the upper devices in order to not destroy the metallization and devices in the bottom layers [116].

An alternative approach is *3D die stacking*, which consists in manufacturing each circuit layer on a separate silicon wafer and then bond either wafers, sliced dies, or sliced dies to unsliced wafers to create 3D Stacked Integrated Circuits (3D-SIC). The inter-die interconnection link can be either physically implemented by micro-bumps and/or with Through-Silicon-Vias (TSVs) [112; 117; 118], or by employing a contactless communication method based on capacitive [119; 120] or inductive coupling [121; 122]. Contactless interconnection schemes face several challenges, the most prominent consisting in ensuring power supply, that slow down their adoption [123]. On the contrary, utilizing TSVs as interconnects between stacked planar dies is perceived as being the most promising approach for the current technology status [124], which is also sustained by the fact that semiconductors industry players have already announced TSV based stacked chip products [125; 126].

A special class of die stacking, known as ”2.5D”, consists in ”stacking” multiple dies side-by-side on a silicon interposer carrier, a technology that is already supported by certain design tools [127]. The interposer is only used to connect the active dies by means of TSVs and wires, which makes 2.5D-SICs easier to manufacture, however they are typically less effective than 3D-SICs in terms of power dissipation in interconnects, bandwidth, and off-chip I/O density [128].

In this dissertation we focus on TSV-based 3D-SICs, which are detailed next.

TSV Based 3D-SICs

The idea of vertically integrating multiple layers of active electronic components into a single circuit through TSVs was envisioned more than 50

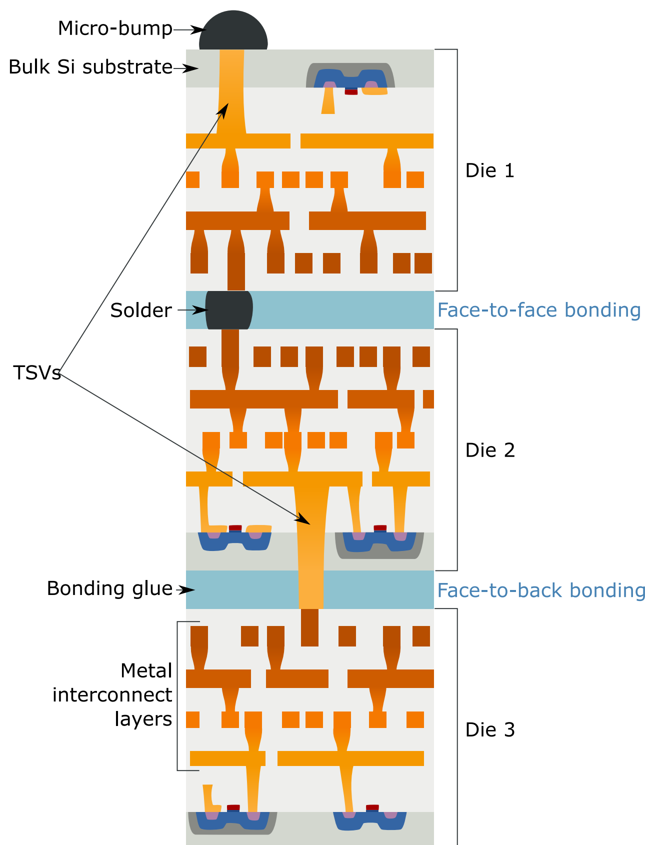


Figure 1.7: TSV based 3D integration.

years ago [129]. It was revisited multiple times since then, when semiconductor engineers thought Moore's law could be stalled by technology issues [129; 130; 131]. In Figure 1.7 is depicted a conceptual 3 layer TSV based 3D integrated circuit that employs the two main stacking techniques, i.e., Face-to-Face (F2F) and Face-to-Back (F2B). In F2F bonding two tiers are stacked such that the very top metal layers are connected. Note that since wafer bonding does not go through a thick buried silicon layer, micro-bumps can be employed as tier interconnects. In F2B bonding device layers are stacked with the top metal layer of one bonded together with the substrate of the other one, requiring in this case TSVs utilization [132]. TSVs are actually holes that traverse the entire silicon substrate and are filled with a conducting material, e.g., copper or tungsten, in order to allow for inter tier communication. TSVs formation can occur at several IC manufacturing stages: via-first, via-middle, via-

last, and via-after-stacking [133]. Via-first TSVs are manufactured prior to the front-end of line, i.e., before transistors are fabricated, and must be filled with doped poly-silicon which has a relatively high resistance. Via-middle TSVs are manufactured between the front-end of line and back-end of line, i.e., before the metal layers are fabricated and typically copper or tungsten is utilized as filling material. Via-last TSVs are manufactured after back-end of line, either prior or post thinning. Compared to via-first and via-middle TSVs, via-last TSVs have the advantage that foundries without TSV processing equipment already may manufacture the whole IC. Finally, in the via-after-stacking approach TSVs are manufactured as the last 3D processing step.

With respect to bonding, there are three methods: Die-to-Die (D2D), Die-to-Wafer (D2W), and Wafer-to-Wafer (W2W) [133]. Although complex, a high alignment accuracy is feasible in D2D and D2W bonding at the cost of a low production throughput. On the other hand, handling very small dies becomes impractical for both D2D and D2W methods, which is simpler for W2W bonding. However, W2W bonding requires stacking dies with same sizes, which makes this approach only suitable for applications with a high degree of regularity, e.g., memories and FPGAs. In addition, W2W stacking negatively impacts the compound yield as it is impossible to prevent good dies stacking over bad dies. This is not the case in D2D and D2W bonding methods, which make use of pre-bond testing prior to the actual stacking to prevent faulty dies from entering the 3D mounting process, which results in an improved compound yield [133].

Having detailed the 3D-SIC technology, next we focus on state of the art research that attempts to exploit its benefits in today's complex memory hierarchy subsystems.

3D Memory Hierarchies

Harvesting 3D technology advantages is possible at different circuit and architectural levels. First, memories as independent circuits can directly benefit from, e.g., the 3D wire length reduction, since their internal regular structure facilitates for easy storage element distribution across multiple layers. Various 3D caches and DRAMs have been proposed in this direction, and even commercial products, such as [125], are available. Additionally, new avenues are also opened at the circuit level in terms of facilitating reliability improvements [134; 135]. Next, the entire multiprocessor memory hierarchy can benefit from 3D technology, since memories themselves can be distributed in the chip 3D space. Thus, technology can ease again the introduction of new archi-

tectural concepts. In the following we review some key strategies applicable at both circuit and architecture levels.

The process of partitioning memories across multiple device layers can take place at different granularities as follows:

- *Array stacking*
  The coarsest 3D memory granularity partitioning takes place at the array level and consists in stacking multiple arrays on the top of each other. We include in here also entire bank stacking, i.e., multiple cell arrays with their afferent address decoders, write drivers, etc. An important overall reduction in wire length is obtained (about 50% for certain configurations), which is translated into significant power and delay gains [136; 137; 138]. The 3D 2 Gb DRAM manufactured by Samsung reported in [125] is based on this bank stacking approach.

- *Intra-array stacking*
  This approach, considered to lead to a true 3D memory [136], breaks the array boundaries and consists in splitting the cells connected to the same bit-line or word-line in groups and placing each group on a different tier [136; 137; 138]. The first approach is also known as *divided-columns* while the latter as *divided-rows*. The peripheral logic, i.e., row decoders, sense amplifiers, column select logic, etc., can be separated from cell arrays and placed on one dedicated layer. This allows for an independent optimization of the peripheral logic for speed, while the cell arrays can be arranged to meet other criteria, e.g., density, footprint, thermal, etc. Research in this area has been performed for both SRAMs [136; 137] and DRAMs [139; 140]. Examples of 3D manufactured SRAMs and DRAMs based on cells stacked on logic are presented in [141]and [142; 143], respectively.

- *Intra-cell (bit) partitioning*
  At this finest granularity level memory cell components are split among one or more layers. The relative small cell size makes the actual implementation of this splitting approach rather impossible for DRAMs. Nevertheless, this option can be feasible for multi-port SRAM arrays, used for example in register file implementations, since they have a larger area than the single port ones, and the cell access transistors could be split among multiple layers [136].

Finally, we add that the previously mentioned partitioning techniques are orthogonal, thus they can be simultaneously utilized in the construction of large

3D memories, e.g., memory cell stacking together with array stacking for the implementation of a large shared cache instance. In addition, it could be desirable to apply different 3D partitioning strategies for different components in the memory hierarchy, e.g., intra-cell partitioning for register files, memory cell stacking for caches, and array stacking for main memory. This results in a complex influence of the partitioning granularity on the rest of the design space parameters, which further increases the existing memory hierarchy design complexity.

With several 3D memories being successfully prototyped and demonstrated, research further focused at the computing system level. As mentioned in Section 1.1.2, in the current 2D setup the main memory is located faraway from the processor, residing somewhere out on the motherboard, and the limited processor package pin number considerably restricts the bandwidth available for the processor. In addition, according to the ITRS, the package pin number is projected to grow only slightly in the coming years and most of these additional pins will be employed for power delivery, instead of data transmission. In an attempt to shrink the processor memory performance gap several solutions have been proposed towards breaking the current processor memory chip boundaries by combining them by means of 3D stacking [72; 144; 145; 146; 147]. Such an approach allows for an increased main memory bus width that can potentially provide enough bandwidth to serve the large amount of requests issued by higher memory hierarchy caches.

It is important to mention that current projections forecast that only up to 1 GB of DRAM can be economically stacked with a processor [148]. This may suffice for implementations targeting certain embedded system applications, but for mainstream computing the die-stacked memory can only serve as an additional last-level cache, or as a subset of the system's main memory [148].

An additional important benefit of 3D integration consists in the possibility of integrating multiple technology types. We detail next other technologies that are being investigated currently in terms of possible CMOS memory replacements and analyze their potential to successfully act as part of hybrid 3D stacked memories.

1.2.2 Emerging Technologies

Currently, a number of new emerging memory technologies, such as Spin Torque Transfer RAM (STT-MRAM) [149], Phase-change RAM (PCRAM) [150], Resistive RAM (RRAM) [151], and Ferroelectric RAMs (FRAM)

[152; 153; 154], are being explored as potential alternatives for existing SRAM and DRAM memories in future computing systems [155]. Such emerging memories should ideally provide SRAM speed, DRAM density, and FLASH non-volatility, without having any other drawbacks, which unfortunately is not the case. As an example, STT-RAM features non-volatility, fast writing/reading speed (< 10 ns), high programming endurance ($> 10^{15}$ cycles), and zero standby power [155]. Thus, utilizing high-density STT-RAM to replace SRAM for caches can reduce the cache miss rate due to larger capacity and thus improve performance. However, the large STT-RAM write latency could induce substantial performance degradation for write-intensive applications. Also, using high density STT-RAM as an additional last cache level before the off-package DRAM can potentially result in substantial average memory access time reductions. However, the cost and complexity of implementing such an approach could be avoided by increasing the currently present last level cache implemented with traditional SRAM technology [155]. Another emerging contendent, the PCRAM, is not fast enough to be employed in caches and does not have the necessary endurance, issues from which RRAM also partly suffers. It can be noticed that the presented emerging technologies stem various design trade-offs [156], but the quest towards an ideal and universal memory technology [157], that would eliminate the need for the current complex memory hierarchies is still on.

Other alternative devices that emerged in the more then Moore context are based on Nano-Electro-Mechanical (NEM) technology. NEM-Relays are electrostatic switches with three terminals that rely on a nano-size mechanical beam to electrically open or close a contact [158]. They feature abrupt switching, due to the electromechanical instability at a certain threshold voltage, and "zero" I_{OFF}, due to the air gap that separates the oxide and the suspended gate in the OFF state. Thus, NEM-Relays potentially offer reduced static energy consumption, however, they require large control voltage, which negatively impacts the dynamic energy consumption. NEM Field Effect Transistors (NEMFETs) are another class of nano-electro-mechanical devices for which the surface potential at the oxide semiconductor interface is controlled by a mechanically moving beam acting as a suspended gate [159]. In this case the current flows through the formed FET channel and not through the beam, as it is the case for NEM-Relays. NEMFETs initial application domain was sensing [160], but due to their extremely low I_{OFF} current several studies proposed them as viable candidates for power saving sleep transistors [161; 162]. Their main drawback is represented by the large switching delay, which for the time being could not be reduced under 1 ns due to the beam movement [78].

Features such as electromechanical hysteresis and sticking make NEM devices attractive for memory applications [163; 164; 165; 166; 167]. The design of a three NEM relay based bitcell architecture is presented in [168]. In [161] the authors replaced SRAM's CMOS inverters with NEMFET equivalent counterparts and reported a substantial 8× leakage reduction at the expense of area and latency overheads. A less power effective approach, which replaces only the nMOS transistors with NEM relays resulting in a diminished area and delay overhead was presented in [169], while, the assessment of such memory cell for thermal management was presented in [170]. A preliminary study of NEMFET-based 1T DRAM memory cells was presented in [171]. In [172] the authors identify the storage capability of a single NEMFET inverter, which opens new opportunities in this area. In view of the above discussion, we believe that NEMFET's inverter potential can be further fostered by its utilization in 3D hybrid memory designs and investigate this research avenue in Chapter 3 of this thesis.

Having introduced the state of the art relevant for our investigations we can further proceed with the formulation of the research questions to be addressed in this dissertation.

1.3 Research Questions

As presented in Section 1.2, 3D technology proved successful in increasing the overall computing systems memory hierarchy performance [72; 173]. 3D memory folding resulted in significant speed-up when applied at the bottom level memory hierarchy components, i.e., main memory and Last Level Caches (LLCs), which previously suffered from long access times due to their large size [136; 137]. Moreover, the traditional memory side narrow interconnection width confined by physical constraints, such as limited package I/O pins, has been significantly increased: 3D integration allowed for a wide memory interconnection width [72; 144; 145; 174]. Since the multi-core architecture paradigm shift, the memory hierarchy was adapted and included shared LLCs in order to efficiently serve multiple simultaneous accesses. LLCs internal banking organization allows them to serve a number of multiple requests in parallel as long as the addresses point to different banks, i.e., there are no bank conflicts. With 3D technology leveraging a wide DRAM interface the memory hierarchy hot spot now seems to be located at the LLCs and, as the core number is constantly scaling up, the concurrent accesses number is also increasing beyond what current LLCs can offer. Thus, it is of great interest to investigate

what 3D can bring in this context and the following question arises:

- **Can 3D TSV-based technology be successfully employed in designing banked multi-port memories with low bank conflict rate and similar to true multi-port designs performance?**

Novel 3D memory designs that go beyond traditional 3D folding should be investigated and evaluated in this context. Ideally, such 3D banked multi-port memories should avoid the burden of extra costs associated with true multi-port memory designs, while not incurring additional performance penalties. In addition, their tiers should be identical in order to reduce manufacturing costs. Due to their memory hierarchy location between main memory and first level caches, such memories should be able to handle different data width accesses, given that the CPU side and long term storage side access granularity requirements are different. With such complex and often contradictory design considerations various trade-offs arise, which should be analyzed. We address this question in Chapter 2 by proposing a versatile polyhedral banked multi-port memory able to ensure a reduced conflict probability.

The next focus is on power dissipation, which became one of the most major processor design concerns. As detailed in Section 1.1, leakage power, once insignificant for the micro-technology generation of ICs, increases abruptly and becomes the total power dissipation dominant component for sub 100 nm CMOS technology nodes [79; 80; 81; 82; 83; 84]. This makes it more challenging for designers and exploring alternative avenues based on, i.e., emerging technologies, becomes of great interest in order to be able to fulfill constrained power requirements, as detailed in Section 1.2. Nano-Electro-Mechanical (NEM) devices are known to exhibit ultra-low leakage currents and abrupt switching, two important characteristics in low power design. However, they are known to suffer from low switching speeds when compared to CMOS devices. In this context we want to investigate if NEMS and CMOS devices can be combined to obtain memory arrays that fully exploit the benefits of both technologies. This leads to the following research question:

- **Can 3D stacked TSV-based technology allow for the successful harvesting of both the appealing NEMS devices low power properties as well as the CMOS devices proven versatility via their hybrid integration?**

To answer this questions we should investigate if such a hybrid 3D memory can exhibit the low power consumption specific to NEMS devices, while re-

taining the quick access times of CMOS only counterparts. Furthermore, we are interested in determining if and by what margin can 3D integration reduce the real estate costs. The potential performance of such hybrid memory arrays for different capacities and memory utilization scenarios from a system level perspective should be evaluated. Last but not least it is of interest to determine at what level in the memory hierarchy can hybrid memories be utilized as CMOS only valid replacements. We address all this issues in Chapter 3, where we propose and evaluate a low power hybrid NEMS-CMOS memory.

Having addressed the performance and power issues, we can shift our attention to dependability aspects. More specifically, to the memory systems trustworthiness in terms of readiness (availability) and continuity in providing correct service. As a first step, it has been suggested that 3D integration brings new opportunities for memory repair [139; 175]. The vertical proximity, which is inherent in 3D stacked memories, can be exploited to expand the memory repairability space with the unused redundant resources from neighboring dies. However, the actual implementations details are still to be explored. Hence, this shapes our next research question as follows:

- **Can state-of-the-art TSV-based 3D integration allow for cost effective memory repair implementations?**

In this context, it is of great interest to investigate what infrastructure has to be embedded into a 3D memory such that spares located on remote dies can be made available to partially faulty memory arrays. This infrastructure has to allow for a user transparent maintenance mechanism able to operate online repairs without affecting the normal memory operation and the eventual incurred penalties are to be determined and evaluated. Various implementation design avenues are to be investigated for both row and column redundancy and their overhead on the memory organization determined. Furthermore, we should clearly identify how reasonable is this overhead for current state of the art TSV technologies. We address this question in Chapter 4 by proposing and evaluating several implementations schemes for 3D memory redundancy based repair.

As mentioned in Section 1.1.4, ECCs supplement redundancy based repair in enabling memory dependability. Conventional memory correction heavily relies on extended Hamming codes to ensure Single Error Correction (SEC) and Double Error Detection (DED), for which 8 check bits are required to protect 64 data bits resulting in an additional 12.5% storage overhead. However, for current technologies a particle hitting a memory array creates disturbances in

multiple physical adjacent cells such that SEC-DED combined with column interleaving techniques are not any longer sufficient in alleviating the large amount of multi-bit upsets. This leads to the following research question:

- **Is it possible to exploit the 3D memories inherent structure to perform multi-error correction within the same redundancy overhead as single error correction, with no incurred performance penalties?**

By investigating the utilization of more complex codes that allow for multi-error correction, e.g., Bose-Chaudhuri-Hocquenghem (BCH), an answer could be provided in this direction. It is worth mentioning that in order to achieve efficient multi-error correction such codes require that the encoding/decoding process is performed on longer codewords than the usual memory read I/O employed for extended Hamming SEC-DED. The circuit complexity required to allow for the longer codewords transfer to the codecs location and back could result in considerable footprint and latency increases. Thus, this operation seems very cost inefficient if is to be handled in traditional 2D circuits, but could potentially be easily performed in 3D by making use of the fast wide-I/O TSV-based vertical access mechanism. A balance has to be found between coding complexity, which has a direct impact on the additional incurred costs, and the desired correction capability. We address this question in Chapter 5.

Another alternative for multi-error correction ECCs is represented by Low Density Parity Check (LDPC) codes, also known as Gallager codes [176]. Proposed as concept in the 1960s, LDPC codes were generally forgotten for decades since it was very difficult to build competitive implementations with the existing technology in those times. In the 1990s their performance was proven substantially better than that of standard convolutional and concatenated codes, exhibiting close to the Shannon performance limit error correction capabilities [177; 178]. At the same time, technology advancements made LDPC implementations possible, and, being rediscovered, LDPCs started to be utilized for sending messages over noisy transmission channels. They proved extremely efficient and are employed in nowadays communication standards such as Wifi (802.11n) and the WiMAX. With respect to memories, in view of their implementation difficulties, translated into long encoding/decoding latencies, LDPC utilization was considered only very recently at very low memory hierarchy levels when Solid State Drives (SSDs) protection switched to LDPC, as reported in [179; 180]. In this context, one can also wonder if LDPC can be employed at higher memory hierarchy levels, such as main memory and caches. Thus, the following research question arises:

- **Can 3D integration enable the utilization of powerful state-of-the-art telecommunication error correction techniques for enhanced protection at higher memory hierarchy levels?**

Due to the large variety of LDPC implementation possibilities a design space exploration should be performed first in order to identify the best LDPC candidates for memory applications. Compared to BCH codes, LDPCs are supposed to provide better correction capabilities at the expense of requiring even larger codewords and longer iterative decoding time. If error correction operations are performed on the fly, a significant memory performance degradation is expected in terms of throughput, latency, and energy consumption. Thus, an adaptive scrubbing error correction modus operandi is a preferred choice, such that the error correction process does not interfere with the normal memory operation. In this context it is of great interest to investigate the eventual penalties incurred for this approach at the system level. We address this question in Chapter 6.

1.4 Dissertation Contributions

In this section, we highlight the main contributions of the research work described in this dissertation, as follows:

- We introduce a novel 3D multi-port memory consisting of multiple identical banks stacked on top of each other, forming a polyhedral structure, that leverages a wide TSV-based data link distributed over the entire memory array. The proposed polyhedral memory supports two orthogonal interfaces: (i) a wide-I/O interface realized vertically through TSV bundles that traverse the entire memory IC stack, and, (ii) a side (horizontal) one, with a lower data width, but with multiple bank-type access ports. The TSV bundles are selectively connected at run-time to the internal data lines in every sub-bank and, in addition to supporting the wide-I/O interface, facilitate horizontal port access to/from a non-local bank (located on a different die) and wide internal data transfers. A reduced rate of bank conflicts is obtained due to the creation of a number of virtual banked ports, greater than the number of physical banks in the memory, while the identical layout structure of all memory dies reduces the manufacturing cost. We study the utilization of our proposal as a last level cache and compared it against traditional banked multi-port

2D implementations, considering relevant metrics: (i) access time, (ii) footprint, and (iii) leakage and dynamic energy consumption per access cycle. The 3D polyhedral cache access time is in general smaller than the one of its planar counterparts, with up to 50% reduction in the best case, while the inherent smaller footprint due to 3D stacking directly influences the active energy consumption, which is almost 8× smaller, for an 8-die 8 MB cache, when compared to planar counterparts. Furthermore, considerable leakage reductions of up to 2× are obtained. Our results indicate that the proposed polyhedral memory clearly outperforms traditional 2D designs for all the considered metrics. In addition, by providing a high bandwidth interconnection with the main memory and ensuring a lower bank conflict rate, the proposed polyhedral memory represents a promising candidate for last level cache implementations.

- We introduce a dual-port 3D-stacked hybrid memory that combines the ultra low power NEMFET capabilities with the mature and fast CMOS technology. The proposed hybrid memory relies on NEMFET based inverters for data storage, and on adjacent CMOS-based logic to allow for quick read/write operations and to ensure data preservation. We study the utilization of our hybrid memory proposal as an alternative to traditional CMOS only SRAMs in L1 and L2 cache implementations of various capacities. About two orders of magnitude static energy reduction were obtained for the proposed hybrid implementations, which is due to NEMFET's extremely low OFF current, and a slightly increased dynamic energy consumption, while an approximately 55% larger footprint was required. The read access time is similar with the SRAM counterparts, while for write operations it is between 2× to 3× higher, depending on the memory size, as it is dominated by the mechanical movement of the NEMFET's suspended gate. In order to determine the potential implications at the system level, we consider as evaluation vehicle a state of the art mobile out-of-order processor core equipped with 32 kB instruction and data L1 caches, and a unified 2 MB L2 cache. We evaluate different scenarios in terms of Instructions-Per-Cycle (IPC) and energy consumption on a set of SPEC benchmarks. Our simulations indicate that for the considered applications, despite of their increased write access time, 3D hybrid L2 cache instances inflict insignificant IPC penalty while providing substantial energy savings of 38% on average, when compared with CMOS only implementations. For L1 instruction caches the IPC penalty is also almost insignificant, while for L1 data caches IPC decreases between 1% to 12% were measured. In addition,

no energy benefits were recorded for any of the L1 instances. Our experiments suggest that the proposed hybrid NEMFET-CMOS memories can be seen as viable candidates to CMOS only alternatives for L2 caches.

- We propose a repair framework for 3D memories based on a provider-consumer pair scheme that allows for inter-die row and column redundant resources sharing. Whenever a memory array has utilized all of its spare resources and is still in need to be able to properly function, it can ”consume” resources from a vertical adjacent memory present in a 3D stack, assuming that it is able to provide them. The proposed infrastructures assure that *consumers* are able to retrieve data from and store data on the *providers* in a transparent manner, i.e., the *providers* function normally, despite their spares being accessed by a neighboring die. Since area represents a sensitive issue in memory design it is important to quantify the feasibility of this approach, considering that it requires additional TSVs and logic. Our analysis in this direction suggests that current TSV dimensions allow inter-die column repair schemes at the expense of reasonable area overhead. For row repair, however, most memory configurations require TSV dimensions to scale down at least with one order of magnitude in order to make this approach a possible candidate for 3D memory repair. We also studied the implications of the proposed 3D repair schemes on the memory access time, with no substantial delay overhead being identified for both inter-die row and column approaches. Our findings indicate that 3D technology facilitates cost effective memory repair implementations.

- We propose two novel error correction mechanisms, specially tailored for polyhedral memories, that ensure effective memory protection against emerging multi-bit upsets, relying on the utilization of BCH multi-error correction codes. The first approach, applied at the die level, consists of placing the required BCH encoders/decoders coplanar with the to be protected memory arrays, and utilizing a codeword length that does not exceed the arrays I/O width. For the second approach, we extend the 3D memory stack with a dedicated tier on which error detection and correction are performed. In addition, we propose to use data from multiple memory array sources, to be able to form longer codewords and employ more powerful codecs. This approach better exploits the polyhedral memories internal organization, which allows for a smooth transfer of the longer codewords to the dedicated die. We mention that the two approaches can work independently or in synergy, which leads to the

creation of a large design space from where the most appropriate scheme can be chosen by balancing the desired error correction capability, while fulfilling different overhead budgets in terms of memory footprint, latency, and energy. To evaluate the error correction capability of our proposals we consider as a case study a 4-die 4-MB polyhedral memory and simulate various data width code implementations. The results indicate that our proposals outperform state of the art Hamming based single error correction schemes in terms of error correction capability, being able to diminish the Word Error Rates (WER) by many orders of magnitude, e.g., WER from 10^{-10} to 10^{-21} are achieved for bit error probabilities between 10^{-4} and 10^{-6}, while requiring less redundancy overhead. We also studied the actual implications of utilizing the proposed approaches in terms of die footprint, latency, and energy consumption. Our findings indicate that by relocating the encoders/decoders from the memory dies to a dedicated one a 13% footprint reduction is obtained. If on the fly encoding/decoding is performed, in addition to the normal memory operations, overheads of up 40% and 83% are to be expected for read and write operations, respectively. In addition, up to 93% energy consumption increase was noticed for read operations, while almost insignificant write energy increases were observed. To reduce the aforementioned overheads we propose the utilization of an adaptive MEC policy that has as main modus operandi a transparent scrubbing based approach, that ensures error detection and correction decoupled from the normal memory operation, and performs on the fly encoding and decoding only when the environmental conditions require so.

- We introduce a novel memory error correction mechanism which, in contrast with state of the art extended Hamming based memory error correction schemes, utilizes LDPC codes due to their close to the Shannon performance limit error correction capabilities. Since for efficient error correction LDPCs require codewords much larger than contemporary memory I/O width, we build our approach on top of a polyhedral memory in order to achieve a cost-effective implementation. In this manner we take advantage of the 3D memories flexible, powerful, and wide data access capabilities, that allow to quickly transfer the large codewords over the 3D memory TSVs to a dedicated die, on which the actual error correction and detection is performed. To make this process transparent to the memory users, e.g., processing cores, we adopt an online memory scrubbing procedure, i.e., at certain time periods the entire memory is scanned and eventual errors are corrected. The scrubbing

policy takes into account possible codeword invalidations that may occur due to the utilization of codewords larger than the memory addressing width. In addition, the error correction operation mode can be adapted to more aggressive environmental conditions, which implies performing LDPC encoding/decoding on the fly with write and read operations, but has a detrimental impact on performance in terms of throughput, latency and energy consumption. To evaluate our proposal we consider 3D memories protected by the proposed LDPC mechanism with various data width codes implementations. We restrict our analysis to what we consider as the best approach for memory protection, i.e., conventional fixed-point implementations of soft-decision decoding with partially or fully parallel architecture under flooded scheduling LDPC decoders. Our simulations indicate that our proposal clearly outperforms state of the art ECC schemes with fault tolerance improvements by a $4710\times$ factor being obtained when compared to extended Hamming ECC. Furthermore, we evaluate instances of the proposed memory concept equipped with different LDPC codecs, implemented on a commercial 40nm low-power CMOS technology and evaluate them on actual memory traces in terms of error correction capability, area, latency, and energy. When compared to state of the art extended Hamming ECC, substantial improvements in error correction capabilities (memory error rates $\alpha < 3 \times 10^{-2}$) are obtained by utilizing our proposed LDPC protection mechanism.

1.5 Dissertation Organization

The remainder of the thesis is structured in 6 chapters as follows:

In **Chapter 2** we introduce and evaluate a novel polyhedral banked multi-port memory design that leverages the TSV-based 3D ICs benefits to provide high bandwidth and close to true multi-port access (with low bank conflict rate). We next asses and compare, by means of simulations, our proposal against state of the art planar memory implementations.

In **Chapter 3** we propose and evaluate a dual port 3D stacked hybrid memory that relies on a NEMFET based inverter for data storage, to benefit from its abrupt switching and hysteresis properties, and on adjacent CMOS logic for data I/O, to benefit from its maturity, versatility, and performance.

In **Chapter 4** we switch to memory repair and address low level issues related to this approach, with a focus on inter-die redundant memory resources

sharing. Several repair schemes are proposed and analyzed in an attempt to determine their potential for current and foreseeable 3D-TSV based technology.

In **Chapter 5** we propose and evaluate a novel multi-error correction scheme for 3D memories, which makes use of Bose-Chaudhuri-Hocquenghem codes.

In **Chapter 6** we propose and evaluate a novel memory multi-error correction mechanism that takes advantage of the 3D memories internal organization capabilities to perform Low Density Parity Check (LDPC) encoding/decoding on very large codewords.

Finally, **Chapter 7** summarizes the thesis accomplishments and identifies significant and promising new opportunities and avenues for follow-up research.

Bibliography

[1] D. Swade, "'it will not slice a pineapple'-charles babbage and the first computer," *IEE Review*, vol. 37, pp. 217–220, June 1991.

[2] W. Aspray, ed., *Computing Before Computers*. Ames, IA, USA: Iowa State University Press, 1990.

[3] H. Aiken, A. G. Oettinger, and T. C. Bartee, "Proposed automatic calculating machine," *IEEE Spectrum*, vol. 1, pp. 62–69, Aug 1964.

[4] R. Serrell, M. M. Astrahan, G. W. Patterson, and I. B. Pyne, "The evolution of computing machines and systems," *Proceedings of the IRE*, vol. 50, pp. 1039–1058, May 1962.

[5] H. H. Goldstine and A. Goldstine, "The electronic numerical integrator and computer (eniac)," *IEEE Annals of the History of Computing*, vol. 18, pp. 10–16, Spring 1996.

[6] W. Stallings, *Computer Organization and Architecture: Designing for Performance (9th Edition)*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 2013.

[7] A. S. Tanenbaum, *Structured Computer Organization (6th Edition)*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 2013.

[8] J. L. Hennessy and D. A. Patterson, *Computer Architecture, Fifth Edition: A Quantitative Approach*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 5th ed., 2011.

[9] J. v. Neumann, "First draft of a report on the edvac," tech. rep., 1945.

[10] M. V. Wilkes, "The edsac computer," in *1951 International Workshop on Managing Requirements Knowledge*, pp. 79–79, Dec 1951.

[11] M. V. Wilkes, "Early computer developments at cambridge: The edsac," *Radio and Electronic Engineer*, vol. 45, pp. 332–335, July 1975.

[12] J. P. Eckert, "A survey of digital computer memory systems," *Proceedings of the IRE*, vol. 41, pp. 1393–1406, Oct 1953.

[13] M. V. Wilkes, "Computers before silicon-design decisions on edsac," *IEE Review*, vol. 36, pp. 429–431, Dec 1990.

[14] F. C. Williams and T. Kilburn, "A storage system for use with binary-digital computing machines," *Electrical Engineers, Journal of the Institution of*, vol. 1949, pp. 112–113, April 1949.

[15] B. Copeland, "The manchester computer: A revised history part 1: The memory," *IEEE Annals of the History of Computing*, vol. 33, pp. 4–21, Jan 2011.

[16] B. Copeland, "The manchester computer: A revised history part 2: The baby computer," *IEEE Annals of the History of Computing*, vol. 33, pp. 22–37, Jan 2011.

[17] E. B. Carne, "Magnetic memory drum design," *Transactions of the American Institute of Electrical Engineers, Part I: Communication and Electronics*, vol. 79, pp. 749–756, Jan 1961.

[18] C. Burton, "The manchester baby reborn [small scale experimental machine]," *IEE Review*, vol. 44, pp. 113–117, May 1998.

[19] J. W. Forrester and R. R. Everett, "The whirlwind computer project," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 26, pp. 903–910, Sep 1990.

[20] S. H. Dodd, H. Klemperer, and P. Youtz, "Electrostatic storage tube," *Electrical Engineering*, vol. 69, pp. 990–995, Nov 1950.

[21] G. E. V. Jr., "How the sage development began," *Annals of the History of Computing*, vol. 7, pp. 196–226, July 1985.

[22] W. N. Papian, "Ferroelectrics and ferromagnetics as memory devices," in *1952 Conference on Electrical Insulation*, pp. 56–56, Oct 1952.

[23] C. Evans, "Conversation: Jay w. forrester," *Annals of the History of Computing*, vol. 5, pp. 297–301, July 1983.

[24] R. Rojas and U. Hashagen, eds., *The First Computers: History and Architectures*. Cambridge, MA, USA: MIT Press, 2002.

[25] E. W. Pugh and W. Aspray, "Creating the computer industry," *IEEE Annals of the History of Computing*, vol. 18, pp. 7–, Summer 1996.

[26] M. V. Wilkes, "Slave memories and dynamic storage allocation," *IEEE Transactions on Electronic Computers*, vol. EC-14, pp. 270–271, April 1965.

[27] L. Edgar, "Method and apparatus for controlling electric currents," Jan. 28 1930. US Patent 1,745,175.

[28] O. Heil, "Improvements in or relating to electrical amplifiers and other control arrangements and devices," 1935. British Patent 439,457 A.

[29] W. F. Brinkman, D. E. Haggan, and W. W. Troutman, "A history of the invention of the transistor and where it will lead us," *IEEE Journal of Solid-State Circuits*, vol. 32, pp. 1858–1865, Dec 1997.

[30] W. Brattain and R. Gibney, "Three-electrode circuit element utilizing semiconductor materials," Oct 1950. US Patent 2,524,034.

[31] J. A. Morton and W. J. Pietenpol, "The technological impact of transistors," *Proceedings of the IRE*, vol. 46, pp. 955–959, June 1958.

[32] J. Hoerni, "Method of manufacturing semiconductor devices," Mar 1962. US Patent 3,025,589.

[33] R. Noyce, "Semiconductor device-and-lead structure," Apr 1961. US Patent 2,981,877.

[34] J. Kilby, "Miniaturized electronic circuits," Jun 1964. US Patent 3,138,743.

[35] C. D. Brady, "Apollo guidance and navigation electronics," *IEEE Transactions on Aerospace*, vol. AS-3, pp. 354–362, June 1965.

[36] W. Aspray, "The intel 4004 microprocessor: what constituted invention?," *IEEE Annals of the History of Computing*, vol. 19, pp. 4–15, Jul 1997.

[37] D. A. Laws, "A company of legend: The legacy of fairchild semiconductor," *IEEE Annals of the History of Computing*, vol. 32, pp. 60–74, Jan 2010.

[38] H. A. Perkins and J. D. Schmidt, "An integrated semiconductor memory system," in *Proceedings of the November 30–December 1, 1965, Fall Joint Computer Conference, Part I*, AFIPS '65 (Fall, part I), (New York, NY, USA), pp. 1053–1064, ACM, 1965.

[39] S. Parke, K. A. Campbell, and C. Mouli, *Memory Technologies*, pp. 171–187. John Wiley and Sons, Ltd, 2013.

[40] R. H. Dennard, "Field-effect transistor memory," Jun 1968. US Patent 3,387,286.

[41] R. H. Dennard, "Evolution of the mosfet dynamic ram: A personal view," *IEEE Transactions on Electron Devices*, vol. 31, pp. 1549–1555, Nov 1984.

[42] "Discussing dram and cmos scaling with inventor bob dennard," *IEEE Design Test of Computers*, vol. 25, pp. 188–191, March 2008.

[43] G. E. Moore, "Intel: Memories and the microprocessor," *Daedalus*, vol. 125, no. 2, pp. 55–80, 1996.

[44] A. Gercekci and A. Krueger, "Trends in microprocessors," in *Solid-State Circuits Conference, 1985. ESSCIRC '85. 11th European*, pp. 233–233i, Sept 1985.

[45] F. Faggin, "The making of the first microprocessor," *IEEE Solid-State Circuits Magazine*, vol. 1, pp. 8–21, Winter 2009.

[46] F. Faggin, M. E. Hoff, S. Mazor, and M. Shima, "The history of the 4004," *IEEE Micro*, vol. 16, pp. 10–20, Dec 1996.

[47] Intel, "Intel's first microprocessor," 2017.

[48] S. P. Morse, B. W. Raveiel, S. Mazor, and W. B. Pohiman, "Intel microprocessors: 8008 to 8086," *Computer*, vol. 13, pp. 42–60, Oct 1980.

[49] Intel, "Intel timeline: a history of innovation," 2017.

[50] G. E. Moore, "Cramming more components onto integrated circuits," in *1975 International Electron Devices Meeting*, vol. 38, April 1965.

[51] Wikipedia, "Instructions per second," 2018.

[52] "From exponential technologies to exponential innovation," tech. rep., Deloitte University Press, 2013.

[53] N. R. Mahapatra and B. Venkatrao, "The processor-memory bottleneck: Problems and solutions," *Crossroads*, vol. 5, Apr. 1999.

[54] W. Wulf and S. A. McKee, "Hitting the memory wall: Implications of the obvious," tech. rep., Charlottesville, VA, USA, 1994.

[55] K. A. El-ayat and R. K. Agarwal, “The intel 80386 - architecture and implementation,” *IEEE Micro*, vol. 5, pp. 4–22, Dec 1985.

[56] P. P. Gelsinger, “Design and test of the 80386,” *IEEE Design Test of Computers*, vol. 4, pp. 42–50, June 1987.

[57] N. Weste and D. Harris, *CMOS VLSI Design: A Circuits and Systems Perspective*. USA: Addison-Wesley Publishing Company, 4th ed., 2010.

[58] R. H. Dennard, F. H. Gaensslen, V. L. Rideout, E. Bassous, and A. R. LeBlanc, “Design of ion-implanted mosfet’s with very small physical dimensions,” *IEEE Journal of Solid-State Circuits*, vol. 9, pp. 256–268, Oct 1974.

[59] D. Foty and G. Gildenblat, “Cmos scaling theory - why our ”theory of everything” still works, and what that means for the future,” in *12th International Symposium on Electron Devices for Microwave and Optoelectronic Applications, 2004. EDMO 2004.*, pp. 27–38, Nov 2004.

[60] M. T. Bohr and I. A. Young, “Cmos scaling trends and beyond,” *IEEE Micro*, vol. 37, pp. 20–29, November 2017.

[61] T. Ghani, M. Armstrong, C. Auth, M. Bost, P. Charvat, G. Glass, T. Hoffmann, K. Johnson, C. Kenyon, J. Klaus, B. McIntyre, K. Mistry, A. Murthy, J. Sandford, M. Silberstein, S. Sivakumar, P. Smith, K. Zawadzki, S. Thompson, and M. Bohr, “A 90nm high volume manufacturing logic technology featuring novel 45nm gate length strained silicon cmos transistors,” in *IEEE International Electron Devices Meeting 2003*, pp. 11.6.1–11.6.3, Dec 2003.

[62] K. Mistry, C. Allen, C. Auth, B. Beattie, D. Bergstrom, M. Bost, M. Brazier, M. Buehler, A. Cappellani, R. Chau, C. H. Choi, G. Ding, K. Fischer, T. Ghani, R. Grover, W. Han, D. Hanken, M. Hattendorf, J. He, J. Hicks, R. Huessner, D. Ingerly, P. Jain, R. James, L. Jong, S. Joshi, C. Kenyon, K. Kuhn, K. Lee, H. Liu, J. Maiz, B. McIntyre, P. Moon, J. Neirynck, S. Pae, C. Parker, D. Parsons, C. Prasad, L. Pipes, M. Prince, P. Ranade, T. Reynolds, J. Sandford, L. Shifren, J. Sebastian, J. Seiple, D. Simon, S. Sivakumar, P. Smith, C. Thomas, T. Troeger, P. Vandervoorn, S. Williams, and K. Zawadzki, “A 45nm logic technology with high-k+metal gate transistors, strained silicon, 9 cu interconnect layers, 193nm dry patterning, and 100Electron Devices Meeting, pp. 247–250, Dec 2007.

[63] C. Auth, C. Allen, A. Blattner, D. Bergstrom, M. Brazier, M. Bost, M. Buehler, V. Chikarmane, T. Ghani, T. Glassman, R. Grover, W. Han, D. Hanken, M. Hattendorf, P. Hentges, R. Heussner, J. Hicks, D. Ingerly, P. Jain, S. Jaloviar, R. James, D. Jones, J. Jopling, S. Joshi, C. Kenyon, H. Liu, R. McFadden, B. McIntyre, J. Neirynck, C. Parker, L. Pipes, I. Post, S. Pradhan, M. Prince, S. Ramey, T. Reynolds, J. Roesler, J. Sandford, J. Seiple, P. Smith, C. Thomas, D. Towner, T. Troeger, C. Weber, P. Yashar, K. Zawadzki, and K. Mistry, “A 22nm high performance and low-power cmos technology featuring fully-depleted tri-gate transistors, self-aligned contacts and high density mim capacitors,” in *2012 Symposium on VLSI Technology (VLSIT)*, pp. 131–132, June 2012.

[64] K. Rupp, “42 years of microprocessor trend data.”

[65] B. A. Nayfeh and K. Olukotun, “A single-chip multiprocessor,” *Computer*, vol. 30, pp. 79–85, Sep 1997.

[66] B. Ackland, A. Anesko, D. Brinthaupt, S. J. Daubert, A. Kalavade, J. Knobloch, E. Micca, M. Moturi, C. J. Nicol, J. H. O’Neill, J. Othmer, E. Sackinger, K. J. Singh, J. Sweet, C. J. Terman, and J. Williams, “A single-chip, 1.6-billion, 16-b mac/s multiprocessor dsp,” *IEEE Journal of Solid-State Circuits*, vol. 35, pp. 412–424, March 2000.

[67] W. Wolf, “Multiprocessor system-on-chip technology,” *IEEE Signal Processing Magazine*, vol. 26, pp. 50–54, November 2009.

[68] B. M. Rogers, A. Krishna, G. B. Bell, K. Vu, X. Jiang, and Y. Solihin, “Scaling the bandwidth wall: Challenges in and avenues for cmp scaling,” in *Proceedings of the 36th Annual International Symposium on Computer Architecture*, ISCA ’09, (New York, NY, USA), pp. 371–382, ACM, 2009.

[69] O. Mutlu and T. Moscibroda, “Parallelism-aware batch scheduling: Enhancing both performance and fairness of shared dram systems,” in *2008 International Symposium on Computer Architecture*, pp. 63–74, June 2008.

[70] E. Ipek, O. Mutlu, J. F. Martínez, and R. Caruana, “Self-optimizing memory controllers: A reinforcement learning approach,” in *2008 International Symposium on Computer Architecture*, pp. 39–50, June 2008.

[71] S. Chen, Y. Hu, Y. Zhang, L. Peng, J. Ardonne, S. Irving, and A. Srivastava, “Increasing off-chip bandwidth in multi-core processors with switchable pins,” in *2014 ACM/IEEE 41st International Symposium on Computer Architecture (ISCA)*, pp. 385–396, June 2014.

[72] G. H. Loh, “3d-stacked memory architectures for multi-core processors,” in *2008 International Symposium on Computer Architecture*, pp. 453–464, June 2008.

[73] T. Kuroda, “Low-power, high-speed cmos vlsi design,” in *Proceedings. IEEE International Conference on Computer Design: VLSI in Computers and Processors*, pp. 310–315, 2002.

[74] P. Zajac, M. Janicki, M. Szermer, C. Maj, P. Pietrzak, and A. Napieralski, “Cache leakage power estimation using architectural model for 32 nm and 16 nm technology nodes,” in *2012 28th Annual IEEE Semiconductor Thermal Measurement and Management Symposium (SEMI-THERM)*, pp. 308–312, March 2012.

[75] S. Borkar, “Exponential challenges, exponential rewards - the future of moore’s law,” in *Int. Conf. on Very Large Scale Integration of System-on-Chip*, p. 2, Dec. 2003.

[76] S. Rodriguez and B. Jacob, “Energy/power breakdown of pipelined nanometer caches (90nm/65nm/45nm/32nm),” in *ISLPED’06 Proceedings of the 2006 International Symposium on Low Power Electronics and Design*, pp. 25–30, Oct 2006.

[77] M. Anis, “Subthreshold leakage current: challenges and solutions,” in *Proceedings of the 12th IEEE International Conference on Fuzzy Systems (Cat. No.03CH37442)*, pp. 77–80, Dec 2003.

[78] “Emerging research devices,” 2009.

[79] V. Sharma, S. Cosemans, M. Ashouei, J. Huisken, F. Catthoor, and W. Dehaene, “Ultra low energy SRAM design for smart ubiquitous sensors,” *IEEE Micro*, vol. 32, no. 5, pp. 10–24, 2012.

[80] H. Kim, J. H. Ahn, and J. Kim, “Exploiting replicated cache blocks to reduce l2 cache leakage in cmps,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 21, pp. 1863–1877, Oct 2013.

[81] S. Manne, A. Klauser, and D. Grunwald, “Pipeline gating: speculation control for energy reduction,” in *Proceedings. 25th Annual International Symposium on Computer Architecture (Cat. No.98CB36235)*, pp. 132–141, Jun 1998.

[82] J. Montanaro, R. T. Witek, K. Anne, A. J. Black, E. M. Cooper, D. W. Dobberpuhl, P. M. Donahue, J. Eno, A. Farell, G. W. Hoeppner, D. Kruckemyer, T. H. Lee, P. Lin, L. Madden, D. Murray, M. Pearce, S. Santhanam, K. J. Snyder, R. Stephany, and S. C. Thierauf, “A 160 mhz 32 b 0.5 w cmos risc microprocessor,” in *1996 IEEE International Solid-State Circuits Conference. Digest of TEchnical Papers, ISSCC*, pp. 214–215, Feb 1996.

[83] J. Chandarlapati and M. Chaudhuri, “Lemap: Controlling leakage in large chip-multiprocessor caches via profile-guided virtual address translation,” in *2007 25th International Conference on Computer Design*, pp. 423–430, Oct 2007.

[84] M. Monchiero, R. Canal, and A. Gonzalez, “Power/performance/thermal design-space exploration for multicore architectures,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 19, pp. 666–681, May 2008.

[85] Y. Ye, S. Borkar, and V. De, “A new technique for standby leakage reduction in high-performance circuits,” in *Symp. on VLSI Circuits*, pp. 40–41, June 1998.

[86] F. Hamzaoglu and M. Stan, “Circuit-level techniques to control gate leakage for sub-100 nm cmos,” in *Proc. of the Int. Symp. on Low Power Electronics and Design*, pp. 60–63, 2002.

[87] J. Tschanz, S. Narendra, Y. Ye, B. Bloechel, S. Borkar, and V. De, “Dynamic sleep transistor and body bias for active leakage power control of microprocessors,” vol. 38, pp. 1838–1845, 2003.

[88] M. Powell, S.-H. Yang, B. Falsafi, K. Roy, and T. N. Vijaykumar, “Gated-vdd: A circuit technique to reduce leakage in deep-submicron cache memories,” in *Proceedings of the 2000 International Symposium on Low Power Electronics and Design*, ISLPED ’00, (New York, NY, USA), pp. 90–95, ACM, 2000.

[89] H. Zhou, M. C. Toburen, E. Rotenberg, and T. M. Conte, “Adaptive mode control: a static-power-efficient cache design,” in *Proceedings 2001 International Conference on Parallel Architectures and Compilation Techniques*, pp. 61–70, 2001.

[90] S. Kaxiras, Z. Hu, and M. Martonosi, “Cache decay: exploiting generational behavior to reduce cache leakage power,” in *Proceedings 28th Annual International Symposium on Computer Architecture*, pp. 240–251, 2001.

[91] K. Flautner, N. S. Kim, S. Martin, D. Blaauw, and T. Mudge, "Drowsy caches: simple techniques for reducing leakage power," in *Proceedings 29th Annual International Symposium on Computer Architecture*, pp. 148–157, 2002.

[92] J. Abella, A. González, X. Vera, and M. F. P. O'Boyle, "Iatac: A smart predictor to turn-off l2 cache lines," *ACM Trans. Archit. Code Optim.*, vol. 2, pp. 55–77, Mar. 2005.

[93] M. Ghosh and H. H. S. Lee, "Virtual exclusion: An architectural approach to reducing leakage energy in caches for multiprocessor systems," in *2007 International Conference on Parallel and Distributed Systems*, vol. 2, pp. 1–8, Dec 2007.

[94] M. Monchiero, R. Canal, and A. Gonzalez, "Using coherence information and decay techniques to optimize l2 cache leakage in cmps," in *2009 International Conference on Parallel Processing*, pp. 1–8, Sept 2009.

[95] H. Kobayashi, I. Kotera, and H. Takizawa, "Locality analysis to control dynamically way-adaptable caches," *SIGARCH Comput. Archit. News*, vol. 33, pp. 25–32, Sept. 2004.

[96] I. Kotera, K. Abe, R. Egawa, H. Takizawa, and H. Kobayashi, "Transactions on high-performance embedded architectures and compilers iii," ch. Power-aware Dynamic Cache Partitioning for CMPs, pp. 135–153, Berlin, Heidelberg: Springer-Verlag, 2011.

[97] N. Cucu-Laurenciu, "Reliability aware computing platforms design and lifetime management."

[98] R. P. Cenker, D. G. Clemons, W. R. Huber, J. B. Petrizzi, F. J. Procyk, and G. M. Trout, "A fault-tolerant 64k dynamic random-access memory," *IEEE Transactions on Electron Devices*, vol. 26, pp. 853–860, Jun 1979.

[99] R. W. Hamming, "Error detecting and error correcting codes," *The Bell System Technical Journal*, vol. 29, pp. 147–160, April 1950.

[100] K. C. Mohr and L. T. Clark, "Delay and area efficient first-level cache soft error detection and correction," in *2006 International Conference on Computer Design*, pp. 88–92, Oct 2006.

[101] N. Seifert, P. Slankard, M. Kirsch, B. Narasimham, V. Zia, C. Brookreson, A. Vo, S. Mitra, B. Gill, and J. Maiz, "Radiation-induced soft error rates of advanced cmos bulk devices," in *2006 IEEE International Reliability Physics Symposium Proceedings*, pp. 217–225, March 2006.

[102] G. Georgakos, P. Huber, M. Ostermayr, E. Amirante, and F. Ruckerbauer, "Investigation of increased multi-bit failure rate due to neutron induced seu in advanced embedded srams," in *2007 IEEE Symposium on VLSI Circuits*, pp. 80–81, June 2007.

[103] A. Dixit and A. Wood, "The impact of new technology on soft error rates," in *2011 International Reliability Physics Symposium*, pp. 5B.4.1–5B.4.7, April 2011.

[104] S. B. Wicker, *Error Control Systems for Digital Communication and Storage*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1994.

[105] K. Osada, Y. Saitoh, E. Ibe, and K. Ishibashi, "16.7 fa/cell tunnel-leakage-suppressed 16 mb sram for handling cosmic-ray-induced multi-errors," in *2003 IEEE International Solid-State Circuits Conference, 2003. Digest of Technical Papers. ISSCC.*, pp. 302–494 vol.1, Feb 2003.

[106] T. Suzuki, Y. Yamagami, I. Hatanaka, A. Shibayama, H. Akamatsu, and H. Yamauchi, "0.3 to 1.5v embedded sram with device-fluctuation-tolerant access-control and cosmic-ray-immune hidden-ecc scheme," in *ISSCC. 2005 IEEE International Digest of Technical Papers. Solid-State Circuits Conference, 2005.*, pp. 484–612 Vol. 1, Feb 2005.

[107] E. Ibe, H. Taniguchi, Y. Yahagi, K. i. Shimbo, and T. Toba, "Impact of scaling on neutron-induced soft error in srams from a 250 nm to a 22 nm design rule," *IEEE Transactions on Electron Devices*, vol. 57, pp. 1527–1538, July 2010.

[108] H. Iwai, "Cmos technology after reaching the scale limit," in *Junction Technology, 2008. IWJT '08. Extended Abstracts - 2008 8th International workshop on*, pp. 1–2, May 2008.

[109] N. Z. Haron and S. Hamdioui, "Why is cmos scaling coming to an end?," in *2008 3rd International Design and Test Workshop*, pp. 98–103, Dec 2008.

[110] J. A. Davis, R. Venkatesan, A. Kaloyeros, M. Beylansky, S. J. Souri, K. Banerjee, K. C. Saraswat, A. Rahman, R. Reif, and J. D. Meindl, "Interconnect limits on gigascale integration (gsi) in the 21st century," *Proceedings of the IEEE*, vol. 89, pp. 305–324, Mar 2001.

[111] J. W. Joyner, P. Zarkesh-Ha, and J. D. Meindl, "A stochastic global net-length distribution for a three-dimensional system-on-a-chip (3d-soc)," in *Proceedings 14th Annual IEEE International ASIC/SOC Conference (IEEE Cat. No.01TH8558)*, pp. 147–151, 2001.

[112] N. Sillon, A. Astier, H. Boutry, L. D. Cioccio, D. Henry, and P. Leduc, "Enabling technologies for 3d integration: From packaging miniaturization to advanced stacked ics," in *2008 IEEE International Electron Devices Meeting*, pp. 1–4, Dec 2008.

[113] E. Beyne, "3d system integration technologies," in *2006 International Symposium on VLSI Technology, Systems, and Applications*, pp. 1–9, April 2006.

[114] Toshiba, "Sip (system in package)."

[115] S. Wong, A. El-Gamal, P. Griffin, Y. Nishi, F. Pease, and J. Plummer, "Monolithic 3d integrated circuits," in *International Symposium on VLSI Technology, Systems and Applications*, pp. 1–4, 2007. Cited by 0031.

[116] L. Brunet, P. Batude, C. Fenouillet-Beranger, P. Besombes, L. Hortemel, F. Ponthenier, B. Previtali, C. Tabone, A. Royer, C. Agraffeil, C. Euvrard-Colnat, A. Seignard, C. Morales, F. Fournel, L. Benaissa, T. Signamarcheix, P. Besson, M. Jourdan, R. Kachtouli, V. Benevent, J. M. Hartmann, C. Comboroure, N. Allouti, N. Posseme, C. Vizioz, C. Arvet, S. Barnola, S. Kerdiles, L. Baud, L. Pasini, C. M. V. Lu, F. Deprat, A. Toffoli, G. Romano, C. Guedj, V. Delaye, F. Boeuf, O. Faynot, and M. Vinet, "First demonstration of a cmos over cmos 3d vlsi coolcube integration on 300mm wafers," in *2016 IEEE Symposium on VLSI Technology*, pp. 1–2, June 2016.

[117] J. H. Lau, “Evolution and outlook of tsv and 3d ic/si integration,” in *2010 12th Electronics Packaging Technology Conference*, pp. 560–570, Dec 2010.

[118] D. Choudhury, “3d integration technologies for emerging microsystems,” in *2010 IEEE MTT-S International Microwave Symposium*, pp. 1–1, May 2010.

[119] K. Kanda, D. D. Antono, K. Ishida, H. Kawaguchi, T. Kuroda, and T. Sakurai, “1.27gb/s/pin 3mw/pin wireless superconnect (wsc) interface scheme,” in *2003 IEEE International Solid-State Circuits Conference, 2003. Digest of Technical Papers. ISSCC.*, pp. 186–487 vol.1, Feb 2003.

[120] S. Mick, L. Luo, J. Wilson, and P. Franzon, “Buried bump and ac coupled interconnection technology,” *IEEE Transactions on Advanced Packaging*, vol. 27, pp. 121–125, Feb 2004.

[121] J. Xu, S. Mick, J. Wilson, L. Luo, K. Chandrasekar, E. Erickson, and P. D. Franzon, “Ac coupled interconnect for dense 3-d ics,” in *2003 IEEE Nuclear Science Symposium. Conference Record (IEEE Cat. No.03CH37515)*, vol. 1, pp. 125–129 Vol.1, Oct 2003.

[122] N. Miura and T. Kuroda, “A 1tb/s 3w inductive-coupling transceiver chip,” in *2007 Asia and South Pacific Design Automation Conference*, pp. 92–93, Jan 2007.

[123] W. R. Davis, J. Wilson, S. Mick, J. Xu, H. Hua, C. Mineo, A. M. Sule, M. Steer, and P. D. Franzon, “Demystifying 3d ics: the pros and cons of going vertical,” *IEEE Design Test of Computers*, vol. 22, pp. 498–510, Nov 2005.

[124] “International technology roadmap for semiconductors,” 2009.

[125] J. S. Kim, C. S. Oh, H. Lee, D. Lee, H. R. Hwang, S. Hwang, B. Na, J. Moon, J. G. Kim, H. Park, J. W. Ryu, K. Park, S. K. Kang, S. Y. Kim, H. Kim, J. M. Bang, H. Cho, M. Jang, C. Han, J. B. Lee, K. Kyung, J. S. Choi, and Y. H. Jun, “A 1.2v 12.8gb/s 2gb mobile wide-i/o dram with 4 x128 i/os using tsv-based stacking,” in *2011 IEEE International Solid-State Circuits Conference*, pp. 496–498, Feb 2011.

[126] M. Santarini, “Stacked and loaded : Xilinx ssi, 28-gbps i/o yield amazing fpgas,” *Xcell Journal*, vol. 74, pp. 8–13, 2011.

[127] Y. Deng and W. P. Maly, “Interconnect characteristics of 2.5-d system integration scheme,” in *Proceedings of the 2001 International Symposium on Physical Design*, ISPD ’01, (New York, NY, USA), pp. 171–175, ACM, 2001.

[128] J. U. Knickerbocker, P. S. Andry, E. Colgan, B. Dang, T. Dickson, X. Gu, C. Haymes, C. Jahnes, Y. Liu, J. Maria, R. J. Polastre, C. K. Tsang, L. Turlapati, B. C. Webb, L. Wiggins, and S. L. Wright, “2.5d and 3d technology challenges and test vehicle demonstrations,” in *2012 IEEE 62nd Electronic Components and Technology Conference*, pp. 1068–1076, May 2012.

[129] J. H. Lau, “Evolution, challenge, and outlook of tsv, 3d ic integration and 3d silicon integration,” in *2011 International Symposium on Advanced Packaging Materials (APM)*, pp. 462–488, Oct 2011.

[130] R. P. Feynman, “The computing machines in the future.”

[131] Y. Akasaka, “Three-dimensional ic trends,” *Proceedings of the IEEE*, vol. 74, pp. 1703–1714, Dec 1986.

[132] Y. Xie and Q. Zou, *3D Integration Technology*, pp. 23–48. New York, NY: Springer New York, 2015.

[133] P. Garrou, C. Bower, and P. Ramm, *3D Integration: Technology and Applications*. Wiley-VCH, 2008.

[134] M. Lefter, G. R. Voicu, M. Taouil, M. Enachescu, S. Hamdioui, and S. D. Cotofana, “Is tsv-based 3d integration suitable for inter-die memory repair?,” in *2013 Design, Automation Test in Europe Conference Exhibition (DATE)*, pp. 1251–1254, March 2013.

[135] S. Safiruddin, D. Borodin, M. Lefter, G. Voicu, and S. D. Cotofana, “Is 3d integration the way to future dependable computing platforms?,” in *2012 13th International Conference on Optimization of Electrical and Electronic Equipment (OPTIM)*, pp. 1233–1242, May 2012.

[136] K. Puttaswamy and G. H. Loh, “3d-integrated sram components for high-performance microprocessors,” *IEEE Transactions on Computers*, vol. 58, pp. 1369–1381, Oct 2009.

[137] Y. F. Tsai, F. Wang, Y. Xie, N. Vijaykrishnan, and M. J. Irwin, “Design space exploration for 3-d cache,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 16, pp. 444–455, April 2008.

[138] P. Reed, G. Yeung, and B. Black, “Design aspects of a microprocessor data cache using 3d die interconnect technology,” in *2005 International Conference on Integrated Circuit Design and Technology, 2005. ICICDT 2005.*, pp. 15–18, May 2005.

[139] R. Anigundi, H. Sun, J. Q. Lu, K. Rose, and T. Zhang, “Architecture design exploration of three-dimensional (3d) integrated dram,” in *2009 10th International Symposium on Quality Electronic Design*, pp. 86–90, March 2009.

[140] U. Kang, H. J. Chung, S. Heo, D. H. Park, H. Lee, J. H. Kim, S. H. Ahn, S. H. Cha, J. Ahn, D. Kwon, J. W. Lee, H. S. Joo, W. S. Kim, D. H. Jang, N. S. Kim, J. H. Choi, T. G. Chung, J. H. Yoo, J. S. Choi, C. Kim, and Y. H. Jun, “8 gb 3-d ddr3 dram using through-silicon-via technology,” *IEEE Journal of Solid-State Circuits*, vol. 45, pp. 111–119, Jan 2010.

[141] A. Zia, P. Jacob, J. W. Kim, M. Chu, R. P. Kraft, and J. F. McDonald, “A 3-d cache with ultra-wide data bus for 3-d processor-memory integration,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 18, pp. 967–977, June 2010.

[142] M. Kawano, S. Uchiyama, Y. Egawa, N. Takahashi, Y. Kurita, K. Soejima, M. Komuro, S. Matsui, K. Shibata, J. Yamada, M. Ishino, H. Ikeda, Y. Saeki, O. Kato, H. Kikuchi, and T. Mitsuhashi, “A 3d packaging technology for 4 gbit stacked dram with 3 gbps data transfer,” in *2006 International Electron Devices Meeting*, pp. 1–4, Dec 2006.

[143] T. Zhang, K. Wang, Y. Feng, X. Song, L. Duan, Y. Xie, X. Cheng, and Y.-L. Lin, “A customized design of dram controller for on-chip 3d dram stacking,” in *IEEE Custom Integrated Circuits Conference 2010*, pp. 1–4, Sept 2010.

[144] C. C. Liu, I. Ganusov, M. Burtscher, and S. Tiwari, “Bridging the processor-memory performance gap with 3d ic technology,” *IEEE Design Test of Computers*, vol. 22, pp. 556–564, Nov 2005.

[145] P. Jacob, A. Zia, O. Erdogan, P. M. Belemjian, J. W. Kim, M. Chu, R. P. Kraft, J. F. McDonald, and K. Bernstein, “Mitigating memory wall effects in high-clock-rate and multicore cmos 3-d processor memory stacks,” *Proceedings of the IEEE*, vol. 97, pp. 108–122, Jan 2009.

[146] D. H. Woo, N. H. Seong, D. L. Lewis, and H. H. S. Lee, “An optimized 3d-stacked memory architecture by exploiting excessive, high-density tsv bandwidth,” in *HPCA - 16 2010 The Sixteenth International Symposium on High-Performance Computer Architecture*, pp. 1–12, Jan 2010.

[147] D. Fick, R. G. Dreslinski, B. Giridhar, G. Kim, S. Seo, M. Fojtik, S. Satpathy, Y. Lee, D. Kim, N. Liu, M. Wieckowski, G. Chen, T. Mudge, D. Sylvester, and D. Blaauw, “Centip3de: A 3930dmips/w configurable near-threshold 3d stacked system with 64 arm cortex-m3 cores,” in *2012 IEEE International Solid-State Circuits Conference*, pp. 190–192, Feb 2012.

[148] G. H. Loh, “Computer architecture for die stacking,” in *Proceedings of Technical Program of 2012 VLSI Technology, System and Application*, pp. 1–2, April 2012.

[149] J. G. Zhu, “Magnetoresistive random access memory: The path to competitiveness and scalability,” *Proceedings of the IEEE*, vol. 96, pp. 1786–1798, Nov 2008.

[150] H. S. P. Wong, S. Raoux, S. Kim, J. Liang, J. P. Reifenberg, B. Rajendran, M. Asheghi, and K. E. Goodson, “Phase change memory,” *Proceedings of the IEEE*, vol. 98, pp. 2201–2227, Dec 2010.

[151] H. S. P. Wong, H. Y. Lee, S. Yu, Y. S. Chen, Y. Wu, P. S. Chen, B. Lee, F. T. Chen, and M. J. Tsai, “Metal-oxide rram,” *Proceedings of the IEEE*, vol. 100, pp. 1951–1970, June 2012.

[152] A. Sheikholeslami and P. G. Gulak, “A survey of circuit innovations in ferroelectric random-access memories,” *Proceedings of the IEEE*, vol. 88, pp. 667–689, May 2000.

[153] H. Shiga, D. Takashima, S. Shiratake, K. Hoya, T. Miyakawa, R. Ogiwara, R. Fukuda, R. Takizawa, K. Hatsuda, F. Matsuoka, Y. Nagadomi, D. Hashimoto, H. Nishimura, T. Hioka, S. Doumae, S. Shimizu, M. Kawano, T. Taguchi, Y. Watanabe, S. Fujii, T. Ozaki, H. Kanaya, Y. Kumura, Y. Shimojo, Y. Yamada, Y. Minami, S. Shuto, K. Yamakawa, S. Yamazaki, I. Kunishima, T. Hamamoto, A. Nitayama, and T. Furuyama, “A 1.6gb/s ddr2 128mb chain feram with scalable octal bitline and sensing schemes,” in *2009 IEEE International Solid-State Circuits Conference - Digest of Technical Papers*, pp. 464–465,465a, Feb 2009.

[154] S. Bagheri, A. A. Asadi, W. Kinsner, and N. Sepehri, “Ferroelectric random access memory (fram) fatigue test with arduino and raspberry pi,” in *2016 IEEE International Conference on Electro Information Technology (EIT)*, pp. 0313–0318, May 2016.

[155] Y. Xie, *Emerging Memory Technologies: Design, Architecture, and Applications*. Springer Publishing Company, Incorporated, 2013.

[156] X. Wu, J. Li, L. Zhang, E. Speight, R. Rajamony, and Y. Xie, “Hybrid cache architecture with disparate memory technologies,” in *Proceedings of the 36th Annual International Symposium on Computer Architecture*, ISCA ’09, (New York, NY, USA), pp. 34–45, ACM, 2009.

[157] J. Åkerman, “Toward a universal memory,” *Science*, vol. 308, no. 5721, pp. 508–510, 2005.

[158] O. Y. Loh and H. D. Espinosa, “Nanoelectromechanical contact switches,” *Nature Nanotechnology*, vol. 7, p. 283–295, Apr 2012.

[159] A. M. Ionescu, V. Pott, R. Fritschi, K. Banerjee, M. J. Declercq, P. Renaud, C. Hibert, P. Fluckiger, and G. A. Racine, “Modeling and design of a low-voltage soi suspended-gate mosfet (sg-mosfet) with a metal-over-gate architecture,” in *Proceedings International Symposium on Quality Electronic Design*, pp. 496–501, 2002.

[160] X. M. H. Huang, M. Manolidis, S. C. Jun, and J. Hone, “Nanomechanical hydrogen sensing,” *Applied Physics Letters*, vol. 86, no. 14, p. 143104, 2005.

[161] H. F. Dadgour and K. Banerjee, “Design and analysis of hybrid nems-cmos circuits for ultra low-power applications,” in *2007 44th ACM/IEEE Design Automation Conference*, pp. 306–311, June 2007.

[162] D. Tsamados, Y. S. Chauhan, C. Eggimann, K. Akarvardar, H. S. P. Wong, and A. M. Ionescu, “Numerical and analytical simulations of suspended gate - fet for ultra-low power inverters,” in *ESSDERC 2007 - 37th European Solid State Device Research Conference*, pp. 167–170, Sept 2007.

[163] D. Tsamados, A. M. Ionescu, Y. Ye, K. Akarvardar, H. S. Philip Wong, E. Alon, and T. J. King Liu, “{ITRS} - nano-electro-mechanical switches,” tech. rep., 2008.

[164] W. Young Choi, H. Kam, D. Lee, J. Lai, and T.-J. K. Liu, “Compact nano-electro-mechanical non-volatile memory (nemory) for 3d integration,” in *Electron Devices Meeting, 2007. IEDM 2007. IEEE International*, pp. 603–606, Dec 2007.

[165] H. Kam, V. Pott, R. Nathanael, J. Jeon, E. Alon, and T.-J. K. Liu, “Design and reliability of a micro-relay technology for zero-standby-power digital logic applications,” in *2009 IEEE International Electron Devices Meeting (IEDM)*, pp. 1–4, Dec 2009.

[166] W. Y. Choi, “Nano-electromechanical (nem) memory cells for highly energy-efficient systems,” in *2011 IEEE Nanotechnology Materials and Devices Conference*, pp. 32–37, Oct 2011.

[167] M. Spencer, F. Chen, C. C. Wang, R. Nathanael, H. Fariborzi, A. Gupta, H. Kam, V. Pott, J. Jeon, T. J. K. Liu, D. Markovic, E. Alon, and V. Stojanovic, “Demonstration of integrated micro-electro-mechanical relay circuits for vlsi applications,” *IEEE Journal of Solid-State Circuits*, vol. 46, pp. 308–320, Jan 2011.

[168] R. Venkatasubramanian, S. K. Manohar, V. Paduvalli, and P. T. Balsara, “Nem relay based memory architectures for low power design,” in *2012 12th IEEE International Conference on Nanotechnology (IEEE-NANO)*, pp. 1–5, Aug 2012.

[169] S. Chong, K. Akarvardar, R. Parsa, J.-B. Yoon, R. Howe, S. Mitra, and H.-S. Wong, "Nanoelectromechanical (nem) relays integrated with cmos sram for improved stability and low leakage," in *IEEE/ACM Int. Conf. on Computer-Aided Design - Digest of Technical Papers*, pp. 478–484, 2009.

[170] X. Huang, C. Zhang, H. Yu, and W. Zhang, "A nanoelectromechanical-switch-based thermal management for 3-d integrated many-core memory-processor system," *IEEE Transactions on Nanotechnology*, vol. 11, pp. 588–600, May 2012.

[171] N. Abele, R. Fritschi, K. Boucart, F. Casset, P. Ancey, and A. M. Ionescu, "Suspended-gate MOSFET: bringing new MEMS functionality into solid-state MOS transistor," in *Proc. of IEEE Int. Electron Devices Meeting*, pp. 479–481, 2006.

[172] K. Akarvardar, C. Eggimann, D. Tsamados, Y. S. Chauhan, G. C. Wan, A. M. Ionescu, R. T. Howe, and H. S. P. Wong, "Analytical modeling of the suspended-gate fet and design insights for low-power logic," *IEEE Transactions on Electron Devices*, vol. 55, pp. 48–59, Jan 2008.

[173] Y. Xie, "Processor architecture design using 3d integration technology," in *2010 23rd International Conference on VLSI Design*, pp. 446–451, Jan 2010.

[174] G. L. Loi, B. Agrawal, N. Srivastava, S.-C. Lin, T. Sherwood, and K. Banerjee, "A thermally-aware performance analysis of vertically integrated (3-d) processor-memory hierarchy," in *2006 43rd ACM/IEEE Design Automation Conference*, pp. 991–996, July 2006.

[175] C.-W. Chou, Y. J. Huang, and J. F. Li, "Yield-enhancement techniques for 3d random access memories," in *Proceedings of 2010 International Symposium on VLSI Design, Automation and Test*, pp. 104–107, April 2010.

[176] R. Gallager, "Low-density parity-check codes," *IRE Transactions on Information Theory*, vol. 8, pp. 21–28, January 1962.

[177] D. J. C. MacKay and R. M. Neal, "Near shannon limit performance of low density parity check codes," *Electronics Letters*, vol. 32, pp. 1645–, Aug 1996.

[178] D. J. C. MacKay, "Good error-correcting codes based on very sparse matrices," *IEEE Transactions on Information Theory*, vol. 45, pp. 399–431, Mar 1999.

[179] K. Zhao, W. Zhao, H. Sun, T. Zhang, X. Zhang, and N. Zheng, "Ldpc-in-ssd: Making advanced error correction codes work effectively in solid state drives," in *Proceedings of the 11th USENIX Conference on File and Storage Technologies*, FAST'13, (Berkeley, CA, USA), pp. 243–256, USENIX Association, 2013.

[180] T. Tokutomi, S. Tanakamaru, T. O. Iwasaki, and K. Takeuchi, "Advanced error prediction ldpc for high-speed reliable tlc nand-based ssds," in *2014 IEEE 6th International Memory Workshop (IMW)*, pp. 1–4, May 2014.

2

A Shared Polyhedral Cache for 3D Wide-I/O Multi-core Computing Platforms

3 **D-Stacked IC** (3D-SIC) based on Through-Silicon-Vias (TSV) is an emerging technology that enables heterogeneous integration and high bandwidth low latency interconnection. Several prior studies have demonstrated that exploiting the vertical proximity of 3D designs, by stacking memory directly on top of a microprocessor, results in significant performance benefits. Moreover, providing unbalanced access, i.e., different line and fetch sizes, to various levels of the memory hierarchy, can further boost performance. In this chapter we propose a 3D novel cache architecture that leverages a wide TSV-based data link distributed on the entire memory array to support two orthogonal interfaces: (i) a vertical one, with a large data width, and, (ii) a side one, with a lower data width, but with more bank-type access ports, which reduces the bank conflict probability. Our simulations indicate that our proposal substantially outperforms planar counterparts in terms of access time, energy, and footprint while providing high bandwidth, low bank conflict rate, and an enriched access mechanism set.

2.1 Introduction

Computing systems are generally utilizing a hierarchical memory organization (depicted in the middle of Figure 2.1), which strives to simultaneously approach the performance of the fastest component, the cost per bit of the cheapest component, and the energy consumption of the most energy-efficient component [1]. Faster, but small memories implemented using a costly technology are present at the top of the hierarchy, closer to the processing units, while slower, but larger inexpensive memories are present at the lower levels.

The interconnection data width between two adjacent levels also determines various trade-offs between performance, energy consumption, and cost. Traditionally, the Last Level Cache (LLC) is linked with a larger width to the upper level, and with a narrower width to the lower level, i.e., main memory. The narrower memory side interconnection widths are due to physical constraints, e.g., limited package I/O pins. With the advent of 3D-Stacked IC (3D-SIC) based on Through-Silicon-Vias (TSV) technology, stacking various memory levels on top of each other enables wider interconnection (wide-I/O) links [2; 3] (see Figure 2.1 left).

To better exploit the thread-level parallelism in multi-core architectures, the lower level shared caches are usually internally organized in banks, that can be accessed in parallel [4]. Thus, it is essentially a multi-port structure which can simultaneously serve multiple requests as long as the addresses point to different banks, i.e., there are no bank conflicts. For a constant number of access ports, the more banks there are the lower the conflict probability is [5]. However, each bank requires its own set of access lines, which negatively affects the area, and subsequently the access time and energy consumption [6]. It has been proven though in [7] that 3D memories can in certain limits alleviate some of this issues due to their reduced footprint (see Figure 2.1 right).

In this chapter, we propose a novel shared cache design for TSV-based wide-I/O multi-core systems with reduced conflict probability. The proposed cache consists of multiple identical banks, stacked on top of each other, forming a polyhedral structure. The interface with the upper level caches (CPU-side) is realized horizontally through one access port on every die, while the wide-I/O interface with the lower levels (memory-side) is realized vertically through TSV bundles that traverse the entire IC stack. The TSV bundles are distributed over the entire cache footprint and selectively connected at run-time to the internal data lines in every sub-bank. In addition to supporting the wide-I/O interface, the TSV bundles facilitate horizontal port access to/from a non-local

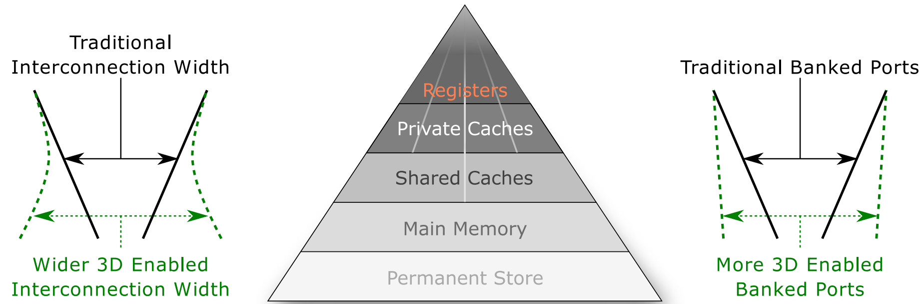


Figure 2.1: Memory hierarchy.

bank (located on a different die) and wide internal cache data transfers. The identical layout structure of all cache dies reduces the manufacturing cost, while the wide-I/O interface increases the communication bandwidth with the lower levels cache or the main memory and provides the means to effectively transfer large amount of data between different banks. In addition, our polyhedral cache has a reduced rate of bank conflicts, due to the creation of a number of virtual banked ports, greater than the number of physical banks in the cache.

To assess the practical implications of our proposal, we compare our novel 3D polyhedral cache against traditional banked multi-port 2D implementations, considering relevant metrics: (i) access time, (ii) footprint, and (iii) leakage and dynamic energy consumption per access cycle. Our results indicate that the 3D polyhedral cache access time is in general smaller than the one of their planar counterparts, with up to 50% reduction in the best case. The inherent footprint reduction directly influences the active energy consumption which in same cases almost reaches the theoretical maximum of, e.g., $8\times$, for an 8-die 8 MB cache. Trade-offs between the wide-I/O interface width and all the other metrics exist: Large widths provide high bandwidth and conflict probability reduction, but increased sub-bank numbers translates into more TSVs and extra decoding logic and wires, which in turn may negatively impact the access time, footprint, and energy cost.

The remainder of the chapter is organized as follows. We detail in Section 2.2 the proposed polyhedral cache internal structure and access scenarios. In Section 2.3 we present an experimental design space exploration for our proposed cache, considering access time, footprint, and energy consumption as relevant metrics. Finally, Section 2.4 concludes this chapter.

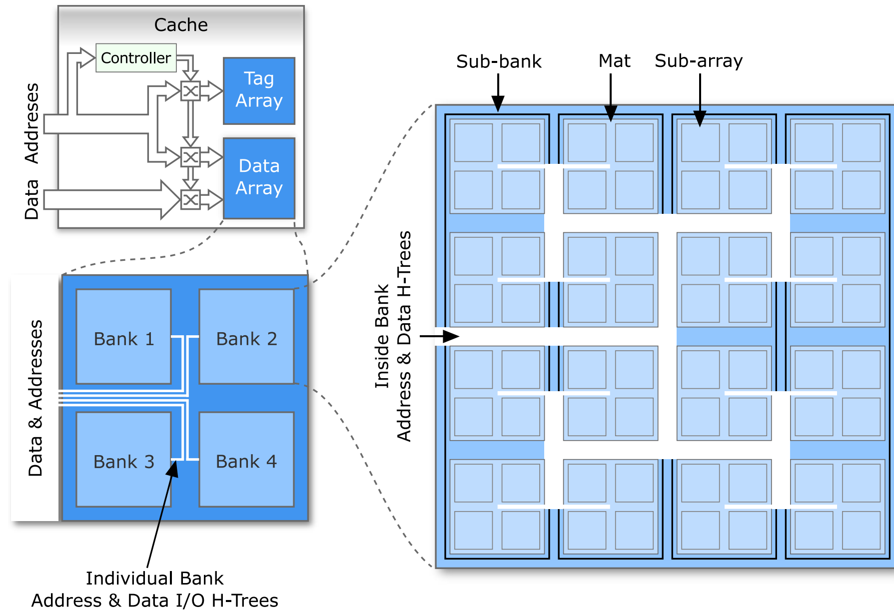


Figure 2.2: Traditional cache and memory array layout.

2.2 Wide-I/O Shared Polyhedral Cache

In this section we introduce the wide-I/O shared polyhedral cache architecture. However, before detailing our proposal, we review for the sake of clarity the required terminology and traditional memory organization layout.

2.2.1 Traditional Cache Design

We consider as a discussion vehicle the banked multiport cache design presented in Figure 2.2, consisting of two separate memory arrays, i.e., tag array and data array, a cache controller, and the required data/addresses crossbar switches. The tag array stores the memory address of the cached blocks, while the data array stores the actual data blocks. Considering a number of input ports and a different/same number of banks, the controller detects the desired bank accesses, arbitrates eventual bank conflicts, and generates the required crossbar switches selection signals.

For both arrays we consider the representative layout employed by the Cacti [8] cache simulator, with a zoom-in into the design abstractions presented in

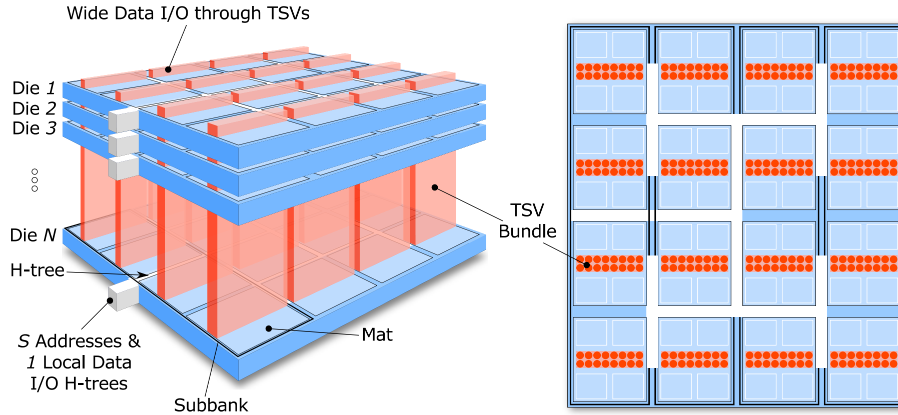


Figure 2.3: Polyhedral memory array.

the figure. At the highest level the address space is split across identical banks, four in this example, with each bank having its own address and data bus, thus allowing for concurrent bank accesses. Each bank is composed of identical sub-banks, again four in this example, with only one being active per access. Further, each sub-bank is partitioned into multiple mats that simultaneously provide parts of the required data (cache block in a cache data array). Finally, each mat is composed of four identical sub-arrays that share predecoding/decoding logic and peripheral circuitry, and which again deliver together the requested data. An H-tree routing distribution network is used to drive addresses and data to/from the banks, and also to/from every mat inside a bank.

2.2.2 Polyhedral Cache Design

In the following we detail the organization of the proposed 3D polyhedral cache. Even though in this work we focus mainly on the cache memory arrays, it is worth noting that the crossbar switches and the controller can also be implemented in 3D technology (either centralized on a separate die, or distributed across several dies).

Our novel cache structure relies on the 3D-stacked memory design presented in Figure 2.3 for both tag and data arrays, where every silicon die is an individually addressable memory bank. Therefore, all addresses, as well as CPU-side data, are routed locally on each die (bank). The memory-side data interface consists of TSV bundles, which traverse all vertically stacked dies through each mat center, connected to the I/O data lines of each sub-array. This creates

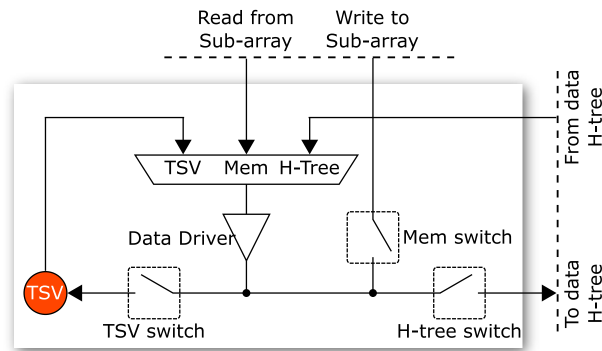


Figure 2.4: Required TSV access logic.

a vertical memory-side wide-I/O interface, having a width equal with the cache block size multiplied by the number of sub-banks. We note that horizontal and vertical data flows can coexist within the polyhedral memory array as long as they do not conflict on TSV bundles and/or sub-banks.

Furthermore, by modifying the sub-array input/output data bit routing logic according to Figure 2.4, we enhance the versatility of the memory array access mechanism such that it allows the following operations:

1. **Local** CPU-side data access when the issue and the storage dies are identical.

2. **Remote** CPU-side data access when the issue die is different from the storage die. In this case both dies must be accessed with the same address, and the TSVs must be connected to the sub-arrays in a complimentary manner: on one die to the sub-arrays inputs, while on the other die one to the sub-arrays outputs.

3. **Wide-I/O** memory-side access performed by simply selecting on which bank, i.e., on which die, to broadcast the address and to link the sub-arrays data I/O to the TSVs.

4. **Inter-die wide transfers** when large data blocks need to be copied from one die (read die) to another die (write die), accomplished by broadcasting the required addresses on both dies. For sub-banks stacked directly on top of each other this is similar to a remote access, with the main difference that data H-tree is not utilized at all, and can be utilized for other non-Wide-I/O accesses. Transfers between different sub-banks however require the data H-tree to be utilized.

Table 2.1: TSV Access Control Signals

Access type		Die	MUX selection	Switches TSV	Switches Mem	Switches H-tree
Local	Read		Mem	OFF	OFF	ON
	Write		H-tree	OFF	ON	OFF
Remote	Read	Issue	TSV	OFF	OFF	ON
		Storage	Mem	ON	OFF	OFF
	Write	Issue	H-tree	ON	OFF	OFF
		Storage	TSV	OFF	ON	OFF
Wide-I/O	Read		Mem	ON	OFF	OFF
	Write		TSV	OFF	ON	OFF
Inter-die		Read	Mem	ON	OFF	OFF
		Write	TSV	OFF	ON	OFF

A maximum of five extra wires are required in the H-tree as control signals to support the above mentioned access policies, as indicated by Table 2.1. In order to maintain a low area overhead, we repurpose the already present sub-array output driver as a signal buffer before or after the TSV to diminish its high capacitive load parasitic effect.

In contrast with a traditional banked memory array layout we choose to extend the number of address H-trees inside every bank. In this manner more remote accesses are possible in parallel on the same bank, which has a direct impact on the reduction of memory conflict probability. Essentially, bank conflicts are reduced to two specific types of sub-bank conflicts: (i) local sub-bank conflicts on the same die, and, (ii) TSV conflicts when sub-banks on different dies are conflicting on the TSV bundles. Thus, the memory conflict probability can be significantly reduced by the proposed cache organization as for an N-die memory with S-address ports per die the maximum success probability is the same as for a memory with $N \times S$ banks, i.e., we can consider that we have a memory with $N \times S$ virtual banks.

2.3 Experimental Evaluation

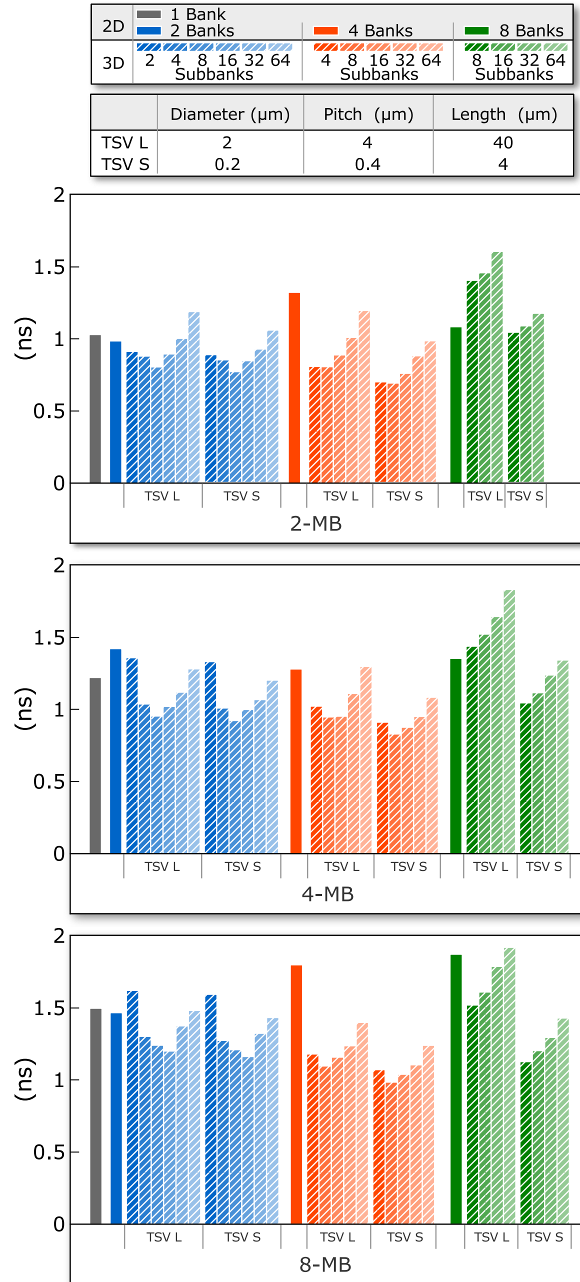
We modified the Cacti 6 [8] simulation model to take into account the extra area of the TSV and control logic and to automatically size the subarray data

drivers to support the capacitive load of a TSV crossing all dies, computed with an electrical equivalent RC model based on the resistance and capacitance equations from [9]. We consider various TSV geometries for which the model was validated against measurements [10]. We measure all relevant metrics, i.e., access time, footprint, dynamic energy, and leakage power of the best 2, 4, and 8 MB caches, considering equal optimization weights for the considered metrics. All instances are 4-way set-associative, with 64B cache block size, and implemented in a high-performance $22nm$ CMOS technology.

We present in Figure 2.5 an access time comparison between banked multi-ported planar and 3D polyhedral caches with 2, 4, and 8 parallel access ports, for different vertical wide-I/O widths (i.e., number of sub-banks). Our experiments indicate that as the TSV is reduced in size all metrics proportionally improve, thus we only plot in Figure 2.5 results for the largest (L) and smallest (S) TSV geometries, with the exact values found in the figure's legend.

We can observe that in general the access time of our proposed 3D caches is better than the one of their planar counterparts, with a 50% reduction in the best case. This is expected, since due to the 3D folding of the cache the footprint is reduced proportionally to the number of dies, as seen in Figure 2.6, and the wires are shorter. The exceptions are explained by the longer wires present in disproportionate sub-banks layout due to the TSVs. Thus, it is only beneficial to have many dies for large caches. The reduction in footprint also leads to a considerable reduction in dynamic energy, with the maximum being close to the theoretical maximum of $8\times$, for an 8-die 8 MB cache. We plot only the read energy in Figure 2.7 since the write energy exhibits similar values. Although the static power is also initially decreasing in Figure 2.8 when more dies are available (because less H-tree signal repetears are needed), the gain is rapidly cancelled out by the extra logic when a lot of TSVs are added.

An important thing to notice from Figures 2.5-2.7 is that trade-offs between the width of the wide-I/O interface and all the metrics exist. Larger widths offer a higher bandwidth and a conflict probability reduction, but the increase in sub-banks translates into more TSVs and extra decoding logic and wires, which in turn negatively impacts the access time, footprint and energy cost. Even though there is an apparent optimum from the access point point of view, the vertical interface width design decision should consider the memory access patterns of the running application, since they influence: (i) the total number of real address conflicts, (ii) the leakage contribution to the total energy consumption, and (iii) the amount of cache pollution [11].



	Diameter (μm)	Pitch (μm)	Length (μm)
TSV L	2	4	40
TSV S	0.2	0.4	4

Figure 2.5: Access time comparison.

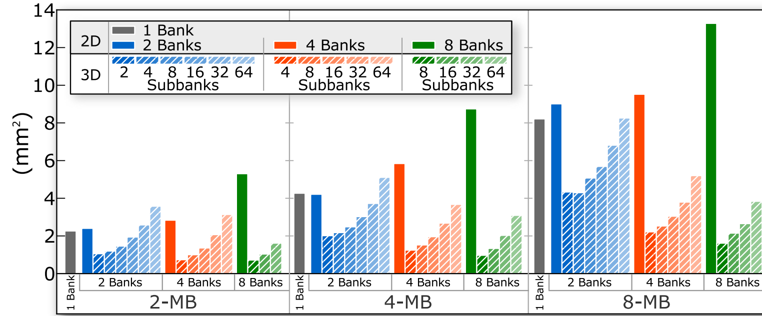


Figure 2.6: Footprint comparison.

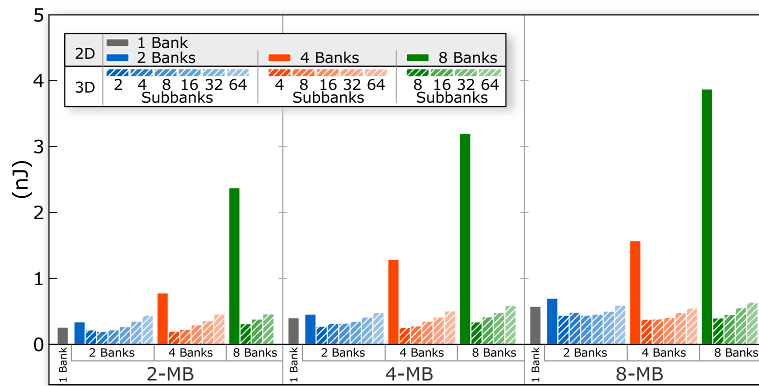


Figure 2.7: Read energy comparison.

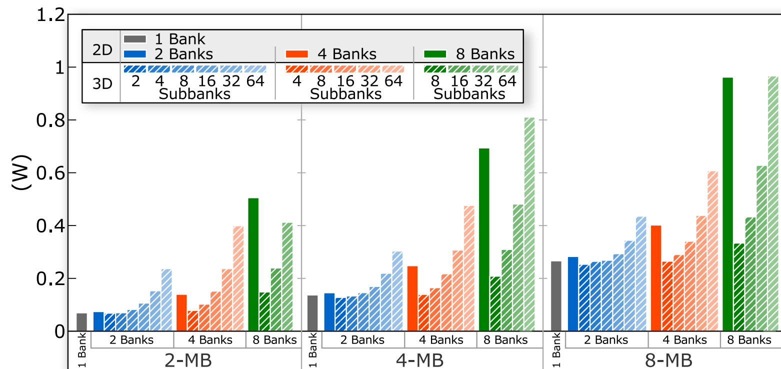


Figure 2.8: Leakage comparison.

2.4 Conclusion

In this chapter, we proposed a novel shared cache design for TSV-based wide-I/O multi-core systems, which consists of multiple identical banks, stacked on top of each other, forming a polyhedral structure. The CPU-side interface is realized horizontally through one access port on every die, while the wide-I/O memory-side interface is realized vertically through TSV bundles that traverse the entire stack. In addition to supporting the wide-I/O interface, the TSV bundles facilitate inter-die wide cache data transfers and the creation of a large number of virtual banked ports. Our simulations indicate that the polyhedral cache outperforms planar counterparts in terms of access time, energy, and footprint, while providing higher bandwidth, lower bank conflict rate, and enriched functionality.

Note. The content of this chapter is based on the following paper:

M. Lefter, G.R. Voicu, S.D. Cotofana, **A Shared Polyhedral Cache for 3D Wide-I/O Multi-Core Computing Platforms**, *IEEE International Symposium on Circuits and Systems (ISCAS)*, Lisbon, Portugal, May 2015.

Bibliography

[1] B. Jacob, S. Ng, and D. Wang, *Memory systems: cache, DRAM, disk*. Morgan Kaufmann Pub, 2007.

[2] C. Liu, I. Ganusov, M. Burtscher, and S. Tiwari, "Bridging the processor-memory performance gap with 3D IC technology," *IEEE Design Test of Computers*, vol. 22, pp. 556–564, Nov 2005.

[3] G. H. Loh, "3d-stacked memory architectures for multi-core processors," in *2008 International Symposium on Computer Architecture*, pp. 453–464, June 2008.

[4] R. Balasubramonian and N. Jouppi, *Multi-Core Cache Hierarchies*. Morgan & Claypool Publishers, 2011.

[5] T. Juan, J. J. Navarro, and O. Temam, "Data caches for superscalar processors," in *International Conference on Supercomputing, ICS '97*, (NY, USA), pp. 60–67, 1997.

[6] N. Muralimanohar, R. Balasubramonian, and N. Jouppi, "Optimizing NUCA organizations and wiring alternatives for large caches with CACTI 6.0," in *MICRO 2007*, pp. 3–14, Dec 2007.

[7] K. Puttaswamy and G. Loh, "3D-integrated SRAM components for high-performance microprocessors," *IEEE Transactions on Computers*, vol. 58, pp. 1369–1381, Oct 2009.

[8] N. Muralimanohar, R. Balasubramonian, and N. P. Jouppi, "CACTI 6.0: A tool to model large caches," tech. rep., HP Laboratories, 2009.

[9] I. Savidis and E. Friedman, "Closed-form expressions of 3-D via resistance, inductance, and capacitance," *IEEE Transactions on Electron Devices*, vol. 56, pp. 1873–1881, Sept 2009.

[10] K. Chen, S. Li, N. Muralimanohar, J.-H. Ahn, J. Brockman, and N. Jouppi, "CACTI-3DD: architecture-level modeling for 3D die-stacked DRAM main memory," in *DATE 2012*, pp. 33–38, March 2012.

[11] D. H. Woo, N. H. Seong, D. Lewis, and H.-H. Lee, "An optimized 3D-stacked memory architecture by exploiting excessive, high-density TSV bandwidth," in *HPCA 2010*, pp. 1–12, Jan 2010.

3

Energy Effective 3D Stacked Hybrid NEMFET-CMOS Caches

In this chapter we propose to utilise 3D-stacked hybrid memories as alternative to traditional CMOS SRAMs in L1 and L2 cache implementations and analyse the potential implications of this approach on the processor performance, measured in terms of Instructions-per-Cycle (IPC) and energy consumption. The 3D hybrid memory cell relies on: (i) a Short Circuit Current Free Nano-Electro-Mechanical Field Effect Transistor (SCCF NEMFET) based inverter for data storage; and (ii) adjacent CMOS-based logic for read/write operations and data preservation. We compare 3D Stacked Hybrid NEMFET-CMOS Caches (3DS-HNCC) of various capacities against state of the art 45 nm low power CMOS SRAM counterparts (2D-CC). All the proposed implementations provide two orders of magnitude static energy reduction (due to NEMFET's extremely low OFF current), a slightly increased dynamic energy consumption, while requiring an approximately 55% larger footprint. The read access time is equivalent, while for write operations it is with about 3 ns higher, as it is dominated by the mechanical movement of the NEMFET's suspended gate. In order to determine if the write latency overhead inflicts any performance penalty, we consider as evaluation vehicle a state of the art mobile out-of-order processor core equipped with 32-kB instruction and data L1 caches, and a unified 2-MB L2 cache. We evaluate different scenarios, utilizing both 3DS-HNCC and 2D-CC at different hierarchy levels, on a set of SPEC 2000 benchmarks. Our simulations indicate that for the considered applications, despite of their increased write access time, 3DS-HNCC L2 caches inflict insignificant IPC penalty while providing, on average, 38% energy savings, when compared with 2D-CC. For L1 instruction caches the IPC penalty is also almost insignificant, while for L1 data caches IPC decreases between 1% to 12% were measured.

3.1 Introduction

With the number of transistors on a single silicon die crossing the one billion threshold, power dissipation became of major concern in processor design, as it directly impacts operating costs and reliability. On the one hand, technology scaling can only marginally reduce power consumption, since the MOSFET threshold voltage (V_T) scalability frontier limits the power supply voltage reduction. On the other hand, leakage power, once insignificant for the micro-technology generation of ICs, increases abruptly and becomes the dominant fraction of the total power dissipation for sub-100 nm technologies [1].

With the advent of emerging nano-technologies, alternative memory arrays have been proposed, which make use of Nano-Electro-Mechanical (NEM) devices, e.g., NEM Field Effect Transistors (NEMFETs) [2],[3], NEM Relays (NEMRs) [4], in conjunction with CMOS devices to substantially reduce their energy consumption. In [5] we proposed a low power NEMFET-based dual-port (one write port and one read port) dual-tier 3D stacked hybrid NEMFET-CMOS memory cell (3D-HdpMC) that combines the appealing ultra-low leakage SCCF NEMFET inverter with the versatility of CMOS technology. We demonstrated that the proposed 3D-HdpMC outperforms the "best of breed", in terms of energy consumption, low-power dual-port SRAM cell (10T-DPMC) [6].

In this chapter we propose the utilization of 3D Stacked Hybrid NEMFET-CMOS Caches (3DS-HNCC) as an alternative to traditional CMOS SRAM caches (2D-CC). We compare 3DS-HNCC of various capacities against state of the art 45 nm low power CMOS SRAM counterparts in terms of relevant metrics for battery operated SoCs, i.e., footprint, delay, and energy consumption. The footprint of 3DS-HNCC is about 55% larger than the one of CMOS only based caches, owing this to NEMFET's mechanical nature and TSV's size. Since an increased footprint impacts the access circuitry energy contribution to the total cache energy, 3DS-HNCC have a dynamic energy increase of 10% and 25% for cache capacities of 32-kB and 2-MB, respectively. However, 3DS-HNCC implementations provide two orders of magnitude static energy reduction, due to NEMFETs extremely low OFF current. 3DS-HNCC's read access time is on average 7% smaller, while its write access time is with about 3 ns higher, as it is dominated by the mechanical movement of the NEMFET's suspended gate.

In order to identify if the write latency overhead inflicts any performance penalty, we consider as evaluation vehicle a state of the art mobile out-of-order

processor core equipped with 32-kB instruction and data L1 caches, and a unified 2-MB L2 cache. We evaluate different scenarios, utilizing 3DS-HNCC caches at different hierarchy levels, on sets of SPEC 2000 [7] benchmarks. We measure an Instructions-per-Cycle (IPC) decrease between 1% to 12% for 3DS-HNCC L1 data caches. However, L1 instruction and L2 caches inflict almost insignificant performance penalty (less than 1% IPC reduction on average). For the considered applications our simulations indicate that, despite of their increased write access time and dynamic energy, L2 3DS-HNCC provide substantial energy savings, i.e., 38% caches energy reduction on the average.

The remainder of the chapter is organized as follows. In Section 3.2 we give a brief presentation of the SCCF NEMFET-based inverter. In Section 3.3 3DS-HNCC are detailed. Next, the section continues with a comparison between 3DS-HNCC and state of the art low power SRAM cache implementations. In Section 3.4 we evaluate the system level implications of utilizing 3DS-HNCC. Finally, Section 3.5 presents our conclusions.

3.2 Low Power Short Circuit Current Free NEMFET Inverter

The Nano-Electro-Mechanical FET (NEMFET), firstly described in [8], is a rather complex device with a 3D geometry and cross-section as presented in Figure 3.1, where H_{BEAM} is the thickness of the suspended gate, W_{BEAM} is the width of the beam, and L_{BEAM} is the length of the beam. In the stable position, there is an *air gap* between the *suspended gate* and the *gate oxide*. If we apply a difference of potential between NEMFET transistor's gate and source, the gate plate, at some point, pulls-in, and touches the oxide. This happens when the transistor is heading towards the inversion region. On the other hand, when the device is about to leave inversion towards depletion, the electrical force gets smaller due to the reduction of the potential difference that generated the force in the first place, and the gate pulls-out to its original in-air position due to the spring force that pulls the gate plate towards its anchors. NEMFET has an extremely low OFF current and exhibits hysteresis as the Pull-In (PI) and Pull-Out (PO) effects occur at different gate voltage values, further denoted as V_{PI} and V_{PO}, respectively [9], [10].

An important power component in any standard CMOS logic gate is the power consumption induced by the short circuit current, which cannot be eliminated. In NEMFET based logic, however, this issue can be alleviated by separately controlling, at design time, the nNEMFET and pNEMFET hysteresis occur-

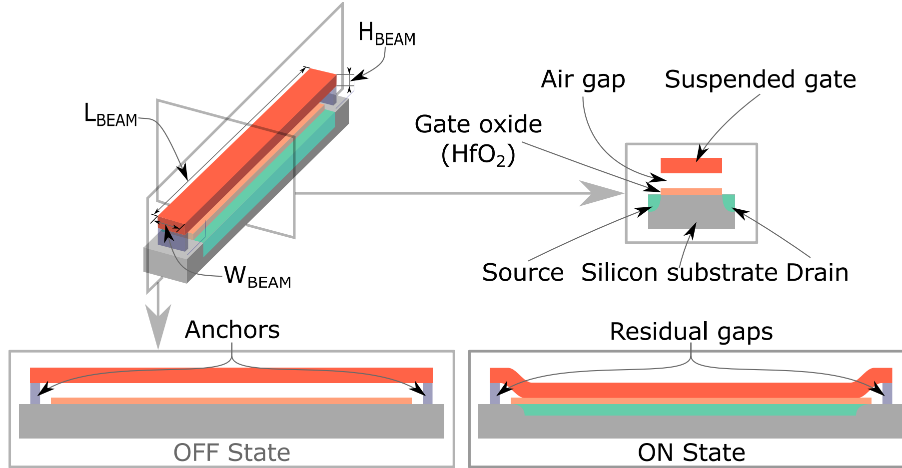


Figure 3.1: NEMFET's geometry [8].

rence, as presented in [11]. The decision on the n/p channel NEMFET sizing (by means of adjusting the suspended beam dimensions) is made such that the following constraints are satisfied: (i) the nNEMFET PI event takes place after the pNEMFET PO event is completed, and (ii) the pNEMFET PI event completes before the nNEMFET PO event starts. From the NEMFET inverter transfer characteristic depicted in Figure 3.2 it can be noticed that when the nNEMFET's beam is pulled-in, the beam of the pNEMFET is pulled-out, and vice versa. In [5] we also addressed the following issues related to NEMFET-based logic: (i) NEMFET based inverter noise margin, (ii) scaling for improved performance, and (iii) energy efficiency against "classic" CMOS technology. We concluded that the NEMFET-inverter *noise margin* is larger than the one of a typical MOSFET-based inverter counterpart, when carefully selecting the geometry of the n/p NEMFET device, such that, the V_{POp} is closer to V_{DD}, and V_{POn} closer to *ground* (see Figure 3.2). NEMFET's Verilog-A compact model [11] was utilized to design a NEMFET based inverter with high *noise margin*, i.e., high hysteresis width. Hence, a $147mV$ (V_{PO}=0.573V V_{PI}=0.720V) hysteresis width was achieved, for $W_{BEAM} = 45nm$, $L_{BEAMn} = 0.9\mu m$, and $L_{BEAMp} = 1\mu m$ which has a 2 orders of magnitude lower leakage, and dynamic energy reduced with 70%, when compared with the CMOS counterpart.

The hysteresis behaviour of the NEMFET inverter makes it suitable for data retention. The storage functionality is the following: the inverter retains the output value unchanged as long as its input is kept at a voltage within the

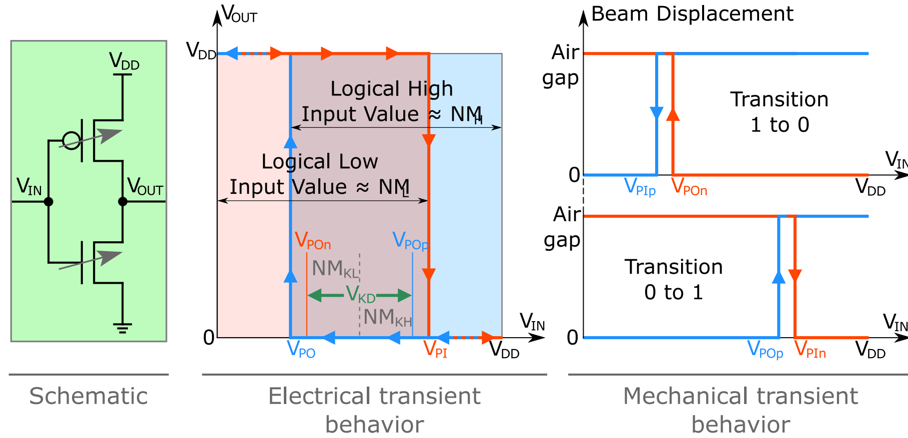


Figure 3.2: NEMFET's inverter schematic and transient behavior.

interval [V_{POp}; V_{POn}] - see also Figure 3.2. We further denote this voltage as V_{KD}. Moreover, Figure 3.2 suggests that for an optimal NEMFET-based memory cell design in terms of *noise margin*, the geometry of the *n* and *p* channel NEMFET should respect the $NM_{KH} = NM_{KL}$ constraint, where: (i) $NM_{KH} = V_{POp} - V_{KD}$, and (ii) $NM_{KL} = V_{KD} - V_{POn}$.

In the next section we detail the utilization of the NEMFET based inverter in the design of low power caches.

3.3 Low Leakage 3D-Stacked Hybrid NEMFET-CMOS Caches

In this section we detail processor cache implementations with 3D-stacked hybrid NEMFET-CMOS memories. First, we present the design and the functionality of such hybrid memories. Next, we focus on cache implementations and provide a comparison against standard CMOS in terms of relevant metrics, i.e., dynamic energy, leakage power, footprint, and access time.

3.3.1 3D-Stacked Hybrid NEMFET-CMOS Memory

The schematic diagram of the 3D-Stacked hybrid NEMS-CMOS memory cell that we proposed in [5] is presented in Figure 3.3, and the storage functionality can be described as follows: (i) the to-be-stored value (0/1) is transmitted at the

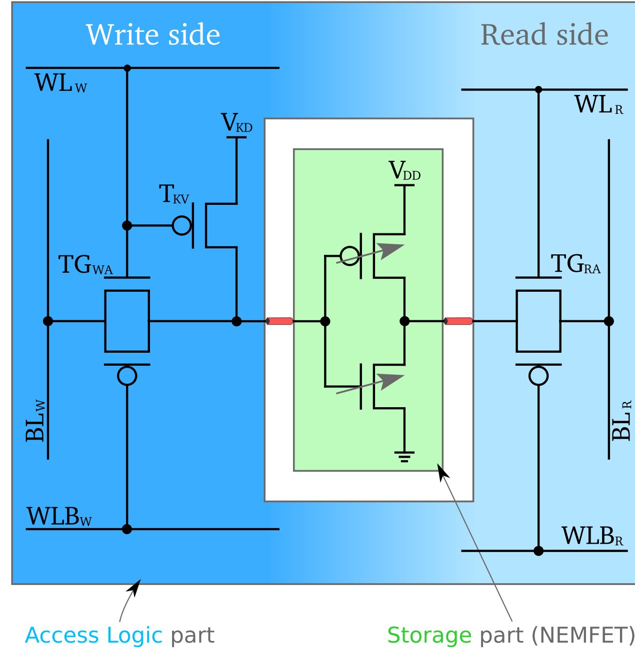


Figure 3.3: Hybrid NEMFET-CMOS memory cell.

input of the NEMFET inverter, (ii) the inverter propagates the inverted value (1/0) at its output, and (iii) the inverter retains the output value unchanged as long as its input is kept at a voltage within the interval $[V_{POpNEMFET}; V_{POnNEMFET}]$, denoted as V_{KD} - see also Figure 3.2. We note that there is a clear and natural separation between the read and the write paths, as also reflected in Figure 3.3. The CMOS logic consists of five transistors: four are forming the transmission gates TG_{WA} and TG_{RA}, which are utilised for write/read operations, and one is utilised for state retention, as briefly explained further on.

In order to write to a memory cell the required value should be present on the write bitline (BL_W). When the write wordline (WL_W) is asserted the transmission gate TG_{WA} opens and the data item reaches the input of the NEMFET-based inverter, which further outputs its complementary value. During the write operation the pMOS transistor T_{KV} is kept closed by the asserted WL_W line. When WL_W is de-asserted the transistor T_{KV} keeps the inverter input at the stable voltage V_{KD}, such that the inverter output value is maintained unchanged. The write access time is mostly determined by the NEMFET switching time, which is rather slow, since due to mechanical considerations

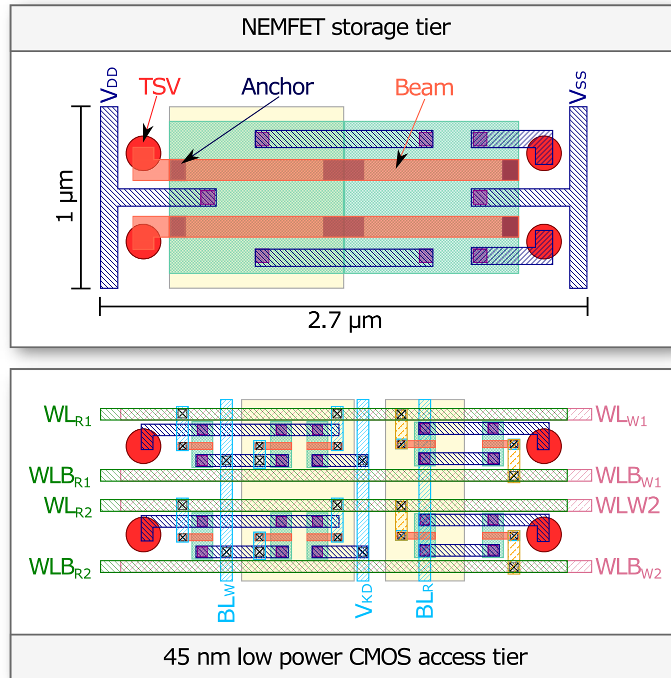


Figure 3.4: Two adjacent hybrid NEMFET-CMOS memory cells layout.

the NEMFET gate requires a certain time interval to stabilize whenever a state change occurs.

In order to read from a memory cell, the read wordline (WL_R) should be asserted. As a result, the transmission gate TG_{RA} opens and frees the stored data item on the read bitline (BL_R). Due to its large dimensions the NEMFET storage inverter behaves as a powerful driver during read, which has a positive impact on the read operation completion time.

We propose to employ TSV-based 3D stacking technology for the final memory structure as it smoothly facilitates the co-integration of NEM and conventional CMOS devices, which, for the time being, appears not to be feasible on the same tier. The proposed memory organization comprises two tiers: the NEMFET-based storage elements reside on the bottom tier, while the CMOS logic required to retrieve, maintain, and write the data is located on the top tier. Tier interconnection is realised through 2 TSVs per memory cell placed one at the input and the other one at the output of the NEMFET inverter, for write and read, respectively (see Figure 3.3).

The 3D-stacked hybrid memory requires the same peripheral interfaces from the circuit designer's point of view. Its clear read and write paths separation does not allow common wordlines and bitlines to be employed for both operations. Thus, the memory can act either as a dual-port memory, and requires dedicated address decoders for read and write operations, or as a single port memory, with a single address decoder and an operation mode signal that selects which wordlines are driven by the decoder's output. Figure 3.4 depicts the layout of two adjacent 3D-Stacked hybrid NEMS-CMOS memory cells that share the same bitline. Considering a TSV with a pitch of 0.4μm [12] and 45nm technology for the CMOS tier, a footprint of $2.7\mu\text{m}\times0.5\mu$m is required for each memory cell.

3.3.2 3D-Stacked Hybrid NEMFET-CMOS Caches

Given the low leakage benefit offered by the 3D-stacked hybrid NEMFET-CMOS memory arrays [5], in the following we address their utilization as caches in the memory hierarchy of a general purpose processor core.

In Table 3.1 we compare caches implemented with 3D-Stacked Hybrid NEMFET-CMOS memories against the most effective, to the best of our knowledge, existing low-power CMOS caches, that employ the 10T non-precharge dual-port SRAM memory cell design introduced in [6]. We consider relevant metrics for battery operated SoCs, i.e., energy, footprint, and delay. The CACTI 6.5 cache memory simulator [13] was utilized to derive the optimal memory partitioning as well as area, access latency, and energy information of peripheral circuitry. A 45 nm low power CMOS technology node was considered, with additional changes being performed to the simulator in order to accommodate the different area required by 10T CMOS and 3D stacked hybrid NEMFET-CMOS cells.

For an accurate access latency, active energy, and leakage characterization the circuits of 10T CMOS and NEMFET-CMOS hybrid memory cell arrays are implemented in a commercial 45 nm low power multi-threshold CMOS technology utilising Cadence Virtuoso [14]. *DC* and *transient* simulations are performed using the Cadence Spectre [14] electric simulator, and Agilent ADS [15] in conjunction with Cadence Spectre, for the CMOS and for the hybrid NEMFET-CMOS memory cell, respectively. For the hybrid cell, we considered the write bitline driver output signal (having as load the bitline and the NEMFET inverter equivalent capacitances) generated by Spectre simulator, as input for the NEMFET inverter simulated with Agilent ADS, by means of the NEMFET's Verilog-A compact model from [11]. The TSV contribution was

Table 3.1: CMOS vs. hybrid NEMFET-CMOS Caches Comparison.

	8-kB 2-way set associative 1 bank		32-kB 4-way set associative 1 bank		512-kB 8-way set associative 2 banks		2-MB 16-way set associative 4 banks	
	10T CMOS	Hybrid	10T CMOS	Hybrid	10T CMOS	Hybrid	10T CMOS	Hybrid
Read energy per access (pJ)	7.29	7.89	11.31	12.57	66.58	76.89	127.34	159.34
Write energy per access (pJ)	8.08	7.64	11.76	12.05	62.64	70.63	123.99	153.78
Leakage (uW)	26.84	0.47	106.69	1.91	1715.48	33.39	6822.38	92.31
Footprint (mm^2)	0.10	0.17	0.30	0.46	5.15	7.84	17.08	27.12
Read access (ns)	1.78	1.55	1.89	1.67	2.66	2.47	3.47	3.30
Write access (ns)	1.41	4.28	1.52	4.36	2.13	5.15	2.88	5.81

also considered, by means of an RLC model from [16], tailored to the TSV diameter from [12; 17]. Moreover, the bitline and wordline RC parasitics, including the bitline coupling capacitance, for each memory array aspect ratio, are extracted from layout, and included in the simulated circuit schematic. Our simulations were performed in typical case conditions, i.e., typical device models, 1.1V supply voltage, and $27°C$.

The footprint of 3D stacked hybrid NEMFET-CMOS caches is with about 55% larger than the one of CMOS only based caches, as it can be observed in Table 3.1. This affects the length, and thus the wire capacitance, with more powerful drivers being necessary, especially for large caches. Therefore, for large capacities, the dynamic energy of the 3D stacked hybrid NEMFET-CMOS caches access logic is greater than for the CMOS counterpart, even though the cache capacity and the technology of the CMOS tier are the same. We can observe in Table 3.1 that for small caches, i.e., 8-kB and 32-kB, the dynamic read/write energies are similar for 3D stacked hybrid NEMFET-CMOS and CMOS only implementations, while for large caches, i.e., 512-kB and 2-MB, there is an increase for the 3D stacked hybrid NEMFET-CMOS implementations of approximately 13% and 25%, respectively.

But NEMFET's larger size is also of advantage, as it makes the inverter-based memory cell a more powerful driver. This has a direct impact on the read access time of the cache, which is on average 7% smaller when compared to CMOS implementations. The write access time of hybrid caches is with about 3 ns higher, as it is dominated by the mechanical movement of the NEMFET's suspended gate. The most important advantage of hybrid caches is given by their almost insignificant leakage, which is two orders of magnitude lower when compared to CMOS only caches.

In the next section we demonstrate that even with a write access time penalty and larger dynamic energy, 3D stacked hybrid NEMFET-CMOS caches provide significant power benefits when compared to CMOS only caches, due to their low leakage advantage.

3.4 System Level Evaluation

In this section we analyse the tradeoffs of utilizing 3D stacked hybrid NEMFET-CMOS caches as replacement for CMOS-only based caches at different levels of the memory hierarchy in a single-core environment. First we describe the simulation environment, i.e., assumed processor configuration model, simulator, benchmarks, and considered evaluation metrics. Provided

Table 3.2: Processor Configuration.

Frequency (GHz)	1.7
L1 size (kB)/associativity	32/4
L1 associativity	4
L1 hit latency CMOS/Hybrid (cycles)	3/3
L1 extra write latency CMOS/Hybrid (cycles)	0/5
L2 size (MB)/associativity	2/16
L2 associativity	16
L2 hit latency CMOS/Hybrid (cycles)	9/9
L2 extra write latency CMOS/Hybrid (cycles)	0/5
Cache line size (bytes)	64
Issue & commit width / Int & FP instruction queues	3/16
Int/FP instruction queue	16
Reorder buffer size	32
Functional units	2 Int, 2 LS, 1 FP

that our proposed hybrid caches have a longer write time when compared to traditional CMOS, we investigate if this penalty propagates at the system level. Next, we focus on energy consumption and determine the benefits that can be achieved by utilizing low power 3D stacked hybrid NEMFET-CMOS caches as replacements for CMOS-only implementations.

3.4.1 Evaluation Methodology and Metrics

We consider a system which mirrors a mobile terminal, based on an ARMv7-A out-of-order processor core detailed in Table 3.2, with a two level cache hierarchy: (i) first level consists of separate instruction and data caches, each 32-kB 4-way set associative, and (ii) second level consists of an unified 2-MB 16-way set associative cache. Cache access cycles were derived from Table 3.1. We simulated the system with a modified gem5 cycle accurate simulator [18], able to accept different cache write and read latencies. In our simulations we employed a set of SPEC CPU2000 [7] benchmarks. To have the exact same workload executed for both CMOS only and hybrid NEMFET-CMOS runs we did not consider benchmarks with non-deterministic behavior. Moreover, some SPEC CPU2000 benchmarks could not be compiled and run in gem5 syscall emulation mode. Nevertheless, the benchmarks set is representative for our comparison.

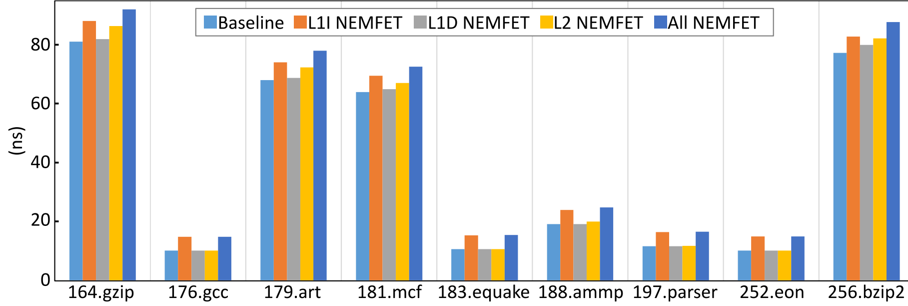


Figure 3.5: L1 instruction cache average miss latencies.

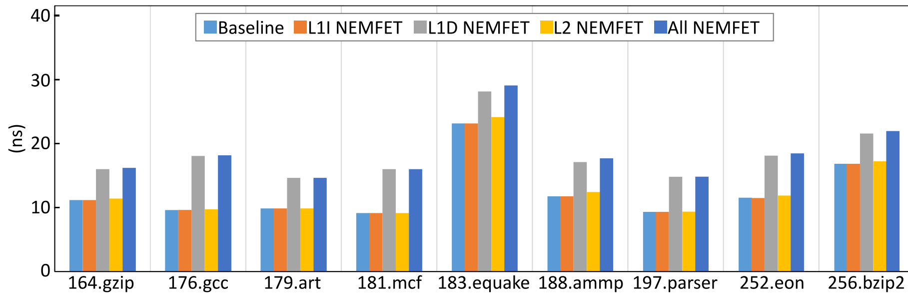


Figure 3.6: L1 data cache average miss latencies.

We considered four different scenarios: (i) **L1I NEMFET**, i.e., the L1 instruction cache is implemented with hybrid NEMFET-CMOS memory, and all other caches with CMOS, (ii) **L1D NEMFET**, i.e., the L1 data cache is implemented with hybrid NEMFET-CMOS memory, and all other caches with CMOS, (iii) **L2 NEMFET**, i.e., L2 cache is implemented with hybrid NEMFET-CMOS memory, and all other caches with CMOS, and, (iv) **All NEMFET**, i.e., both instruction/data L1 and L2 caches are implemented with hybrid NEMFET-CMOS memory. These four scenarios were compared against a **Baseline** design, i.e., all caches implemented with CMOS memories.

To accurately measure the impact on performance of a longer cache write latency, we utilize in our evaluation the cache average miss latency and Instructions-per-Cycle (IPC) architecture metrics. The number of read/write cache accesses is computed based on the cache hits and misses statistics extracted from the gem5 simulator, and further utilized to determine the energy for each application.

3.4.2 Analysis

We start the analysis by observing the average miss latencies for all private L1 instruction and data caches, depicted in Figure 3.5 and Figure 3.6, respectively. It is clear that the extra write latency of NEMFET-CMOS based caches affects the miss latencies, which increase with 24% and 47% for **L1I NEMFET** and **L1D NEMFET**, respectively.

Figure 3.7 presents the IPC reduction of the four scenarios relative to the **Baseline**, for the considered set of benchmarks (the higher the value, the greater the negative performance impact). It is apparent from the figure that the **L1I NEMFET** scenario exhibits insignificant IPC reduction, i.e., less than 0.5%, for almost all benchmarks. There are however two exceptions, i.e., 176.*gcc* and 252.*eon*. Even though for these two benchmarks, a similar behavior in terms of average miss latencies is noticed (see Figure 3.5), greater IPCs losses were obtained, i.e., 0.85% and 2.87%, respectively. Still, also these IPC reduction values are relatively low, translating in a low performance degradation. Similarly, almost insignificant IPC reduction can be noticed in Figure 3.7 for the **L2 NEMFET** scenario.

For the **L1D NEMFET** scenario, IPC reductions between 1.08-12.21% are observed, when compared to the **Baseline**. This can be explained as follows: (i) L1 data cache is more often accessed in comparison with L1 instruction (data are read/written by he processor, while instructions are only read), (ii) L1 data cache is more often accessed in comparison with L2, being closer to the processor, (iii) miss rates for L1 data are greater than in other caches, and, (iv) larger L1 data average miss penalties when compared to the other caches are observed (see Figure 3.6) due to the extra write latency.

Finally, the largest IPC reductions are obtained for the **All NEMFET** scenarios. This is natural, since all the caches are affected by the extra write latencies, which impact the overall performance. On the other hand, **L1I NEMFET** and **L2 NEMFET** scenarios exhibit almost insignificant IPC reductions, possibly due to the program behavior for the former and due to the less frequent accesses for the latter.

Figure 3.8 depicts the relative energy differences from the **Baseline** of the four scenarios. Positive values denote the energy reduction percentages, while negative values stand for energy increases. Due to the low leakage advantage of NEMFET-CMOS memories, an important energy reduction, i.e., 38% on the average, is obtained for the **L2 NEMFET** scenario. This happens because: (i) L2 is larger, thus its energy is dominant in the total cache hierarchy energy,

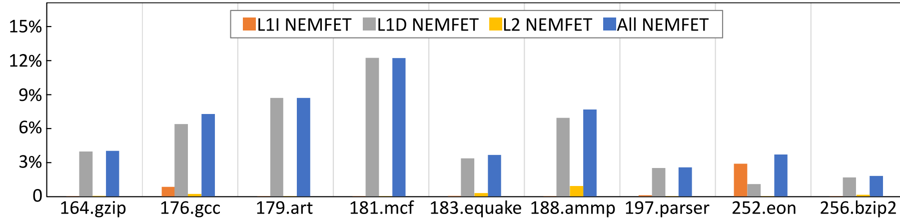


Figure 3.7: IPC reduction relative to **baseline** (lower is better).

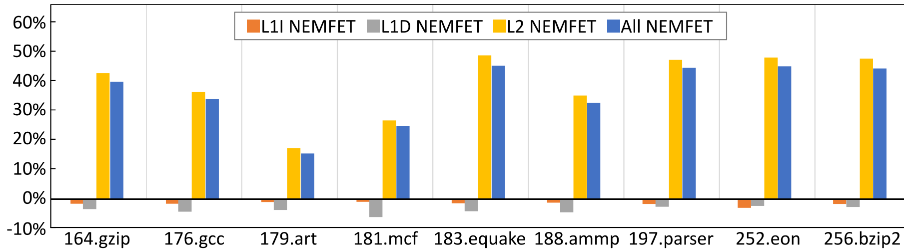


Figure 3.8: Energy difference relative to **baseline** (higher is better).

and, (ii) much fewer accesses in L2 than in L1 caches generate an increase in the contribution of L2 static energy to the total energy. For **L1I NEMFET** and **L1D NEMFET** scenarios an energy increase is observed in all cases, with maximums of 3% and 5.58%, respectively. This is related to the fact that both L1 caches are very often accessed, thus, they have a high activity factor, leading to a higher dynamic energy contribution. Moreover, NEMFET-CMOS caches exhibit a higher dynamic energy when compared to their CMOS counterparts (see Table 3.1). An important energy reduction, i.e., 35% on the average, is obtained for the **All NEMFET** scenario, since it cumulates the effects of all scenarios, thus the energy benefits in L2 caches compensate the energy losses in L1 caches.

3.5 Conclusion

In this chapter we proposed to replace the traditional CMOS based processor caches with energy effective 3D-Stacked Hybrid NEMFET-CMOS ones. This solution provides two orders of magnitude cache static energy reduction, due to the NEMFET's extremely low OFF current, albeit with a slightly increased dynamic energy consumption, an approximately 55% larger footprint, and a longer write access time. We compared the performance of 3D-Stacked Hy-

brid NEMFET-CMOS against 45 nm low power CMOS SRAM based cache embodiments at different levels in the memory hierarchy. In order to find out if the write latency overhead inflicts any performance penalty, we considered as evaluation vehicle a state of the art mobile out-of-order processor core. Our simulation on a set of SPEC 2000 benchmarks proved that the extra write latency impact on the overall system performance is rather low or even negligible. Next we evaluated the energy benefits of utilizing 3D-Stacked Hybrid NEMFET-CMOS. Our simulations indicate that, in spite of their increased write access time and dynamic energy, 3D Stacked Hybrid NEMFET-CMOS L2 caches provide substantial energy savings, i.e., 38% total caches energy reduction on the average.

Note. The content of this chapter is based on the following paper:

M.Lefter, M. Enachescu, G.R. Voicu, S.D. Cotofana, **Energy Effective 3D Stacked Hybrid NEMFET-CMOS Caches**, *ACM/IEEE International Symposium on Nanoscale Architectures (NANOARCH)*, Paris, France, March 2014.

Bibliography

[1] S. Borkar, “Exponential challenges, exponential rewards - the future of Moore’s law,” *VLSI-SOC*, 2003.

[2] N. Abele, A. Villaret, A. Gangadharaiah, C. Gabioud, P. Ancey, and A.-M. Ionescu, “1t mems memory based on suspended gate mosfet,” in *International Electron Devices Meeting (IEDM)*, pp. 1–4, Dec 2006.

[3] H. Dadgour and K. Banerjee, “Hybrid nems-cmos integrated circuits: A novel strategy for energy-efficient designs,” *Computers Digital Techniques, IET*, vol. 3, pp. 593–608, November 2009.

[4] R. Venkatasubramanian, S. Manohar, V. Paduvalli, and P. Balsara, “Nem relay based memory architectures for low power design,” in *IEEE Conference on Nanotechnology (IEEE NANO)*, pp. 1–5, Aug 2012.

[5] M. Enachescu, M. Lefter, G. Voicu, and S. D. Cotofana, “Low-leakage 3d stacked hybrid nemfet-cmos dual port memory,” *IEEE Transactions on Emerging Topics in Computing*, vol. PP, no. 99, pp. 1–1, 2016.

[6] H. Noguchi, S. Okumura, Y. Iguchi, H. Fujiwara, Y. Morita, K. Nii, H. Kawaguchi, and M. Yoshimoto, “Which is the best dual-port sram in 45-nm process technology? - 8t, 10t single end, and 10t differential -,” in *IEEE International Conference on Integrated Circuit Design and Technology and Tutorial (ICICDT)*, pp. 55–58, June 2008.

[7] J. L. Henning, “SPEC CPU2000: measuring CPU performance in the new millennium,” *Computer*, vol. 33, pp. 28–35, Jul 2000.

[8] A.-M. Ionescu, V. Pott, R. Fritschi, K. Banerjee, M. Declercq, P. Renaud, C. Hibert, P. Fluckiger, and G.-A. Racine, “Modeling and design of a low-voltage soi suspended-gate mosfet (sg-mosfet) with a metal-over-gate architecture,” in *International Symposium on Quality Electronic Design (ISQED)*, pp. 496–501, 2002.

[9] K. Akarvardar, C. Eggimann, D. Tsamados, Y. Singh Chauhan, G. C. Wan, A. M. Ionescu, R. T. Howe, and H. S. P. Wong, “Analytical modeling of the suspended-gate FET and design insights for low-power logic,” *IEEE Tran. on Electron Devices*, vol. 55, no. 1, pp. 48–59, 2008.

[10] Enachescu, A. van Genderen, S. Cotofana, A. Ionescu, and D. Tsamados, “Can SG-FET replace FET in sleep mode circuits?,” in *International ICST Conference on Nano-Networks*, (Luzern, Switzerland), pp. 99–104, October 2009.

[11] M. Enachescu, M. Lefter, A. Bazigos, A. Ionescu, and S. Cotofana, “Ultra low power NEMFET based logic,” in *IEEE International Symposium on Circuits and Systems (IS-CAS)*, pp. 566–569, May 2013.

[12] A. Topol, D. La Tulipe, L. Shi, S. M. Alam, D. Frank, S. Steen, J. Vichiconti, D. Posillico, M. Cobb, S. Medd, J. Patel, S. Goma, D. Dimilia, M. T. Robson, E. Duch, M. Farinelli, C. Wang, R. Conti, D. M. Canaperi, L. Deligianni, A. Kumar, K. Kwietniak, C. D’Emic,

J. Ott, A. M. Young, K. Guarini, and M. Ieong, “Enabling soi-based assembly technology for three-dimensional (3d) integrated circuits (ics),” in *IEEE International Electron Devices Meeting (IEDM) Technical Digest.*, pp. 352–355, Dec 2005.

[13] N. Muralimanohar, R. Balasubramonian, and N. P. Jouppi, “CACTI 6.0,” tech. rep., 2007.

[14] “Cadence Design Systems.” http://www.cadence.com/us/pages/default.aspx, 2011.

[15] “Advanced design system (ADS) | agilent.” http://www.agilent.com/, 2012.

[16] H. Chaabouni, M. Rousseau, P. Leduc, A. Farcy, R. El Farhane, A. Thuaire, G. Haury, A. Valentian, G. Billiot, M. Assous, *et al.*, “Investigation on TSV impact on 65nm CMOS devices and circuits,” in *IEEE IEDM*, 2010.

[17] A. Topol *et al.*, *3D Fabrication Options for High-Performance CMOS Technology*, vol. Wafer Level 3-D ICs Process Technology. Springer, 2008.

[18] N. Binkert, B. Beckmann, G. Black, S. K. Reinhardt, A. Saidi, A. Basu, J. Hestness, D. R. Hower, T. Krishna, S. Sardashti, R. Sen, K. Sewell, M. Shoaib, N. Vaish, M. D. Hill, and D. A. Wood, “The gem5 simulator,” *SIGARCH Comput. Archit. News*, vol. 39, pp. 1–7, Aug. 2011.

4

Is TSV-based 3D Integration Suitable for Inter-die Memory Repair?

3D Stacked Integrated Circuit (3D-SIC) technology is emerging as a promising avenue to sustain the continuous need in increased device density and fast on-chip communication. Vertical proximity, inherent in 3D stacked memories, can be exploited to expand the memory reparability space with the unused rows/columns spares from neighboring dies. In previous work, 3D memory repair has been proposed, however, without proper hardware designs. In this chapter we address lower level issues related to 3D inter-die memory repair in an attempt to evaluate the actual potential of this approach for current and foreseeable technology developments. We propose several implementation schemes both for inter-die row and column repair and evaluate their impact in terms of area and delay. Our analysis suggests that current state-of-the-art TSV dimensions allow inter-die column repair schemes at the expense of reasonable area overhead. For row repair, however, most memory configurations require TSV dimensions to scale down at least with one order of magnitude in order to make this approach a possible candidate for 3D memory repair. We also performed a theoretical analysis of the implications of the proposed 3D repair schemes on the memory access time, which indicates that no substantial delay overhead is expected and that many delay versus energy consumption tradeoffs are possible.

4.1 Introduction

Recent enhancements in Integrated Circuits (ICs) manufacturing process enable the fabrication of three dimensional stacked ICs (3D-SICs) based on Through-Silicon-Vias (TSVs) as die-to-die (D2D) interconnects, which further boost the trends of increasing transistor density and performance. 3D-SIC is an emerging technology, that, when compared with planar ICs, allows for smaller footprint, heterogeneous integration, higher interconnect density between stacked dies, and latency reduction mostly due to shorter wires [1].

3D memories have been proposed ever since the technology was introduced, one of the reason being their regular structure that allows them to be easily folded accross bitlines/wordlines and spread over multiple layers in a 3D embodiment [2]. Moreover, the typical area of a System on a Chip (SoC) is memory dominated, and, as the ITRS roadmap predicts that the trend of memory grow continues [3], it is expected that memories will play a critical role in 3D-SICs as most of the layers in the stack are likely to be allocated for storage.

As technology keeps shrinking towards meeting the requirements of increased density, capacity, and performance, IC circuits, memory arrays included, are more prone to degradation mechanism [4], and different sorts of defects during the manufacturing process [5]. In addition, the utilization of the still in its infancy 3D stacking technology increases the risk of low yield. To deal with this issue several works proposed inter-die memory repair, i.e., sharing redundant elements (rows/columns) between layers, in an attempt to increase the compound yield of memories [6; 7; 8; 9; 10; 11].

Up until now, all the work targeting inter-die memory repair primarily discussed the idea in principle, with no real implications being studied. The proposed approaches have been only evaluated via fault injection simulations and the obtained repair rate improvements form an upper bound. In order to achieve inter-die repair, a certain infrastructure has to be embedded into the memory such that spares can be made available to memory arrays in need that are located on remote dies. The added infrastructure must not affect the normal operation of the memory and may incur certain penalties in terms of area and/or delay which have not be studied.

In this chapter we build upon previous work proposals and we further investigate the real implications of inter-die memory repair based on redundancy sharing. We first provide a classification of the possible access scenarios to memory arrays stacked in a 3D memory cube. Next, we propose several implementation schemes both for inter-die row and column repair in which we

detail the circuit infrastructure required to support these access scenarios. For each scheme we propose the infrastructure, highlight its advantages and disadvantages, and discuss its impact on memory area and delay.

The area overhead is mostly dependent on the TSV size rather than on the extra logic. From our analysis it results that current state-of-the-art TSV dimensions allow inter-die column repair schemes with reasonable area overhead. For row repair, however, most memory configurations require TSV dimensions to scale down with at least one order of magnitude to make this approach applicable in practical 3D memory systems. We also performed a theoretical analysis of the implications of the proposed 3D repair schemes on the memory access time. Assuming a 20ps TSV delay our analysis indicates that for row repair the overhead is negligible and for column repair it can be in the same order of magnitude. This indicates that no substantial delay overhead is expected and that many delay versus energy consumption tradeoffs are possible.

The remaining of the chapter is organized as follows. Section 4.2 briefly describes general memory repair techniques and related work regarding 3D memory repair. Section 4.3 defines the 3D memory repair architecture and the associated framework for inter-die redundancy. Section 4.4 introduces the circuit infrastructure necessary to support inter-die memory repair. Section 4.5 considers various trade-offs and cost overhead in terms of area and delay. Finally, Section 4.6 concludes the chapter.

4.2 Redundancy Based Memory Repair

State of the art memory repair relies on addition of several spare[1] resources to the memory arrays, a rather easy task facilitated by memories high regular structure. These resources do not affect the interface or capacity of the memory, but can be later on utilized to substitute memory cells affected by, usual, permanent errors. Therefore, a defect tolerant redundancy based memory directly sustains the yield improvement by reducing the number of discarded memory devices because of a few faulty cells [12]. Prior to allocating spare resources, faulty cells in the memory have to be identified, an operation that is usually performed by a Built-In Self-Test (BIST) circuit. Subsequently, a repair allocation algorithm determines the spare element that substitutes the faulty element. This repair phase is performed by a Built-In Self-Repair (BISR) circuit [13]. Based on the physical placement of the spare elements

[1]We use spare/redundant, rows/wordlines and columns/bitlines interchangeably in the text.

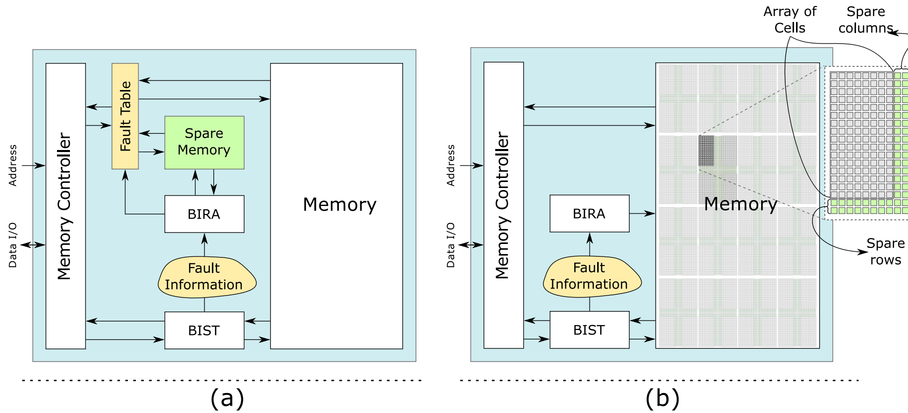


Figure 4.1: Memory redundancy with external (a) and internal (b) spares.

we can broadly distinguish two types, that are not excluding one another, of memory redundancy:

- *External redundancy*, in which a special smaller memory, external to the initial one is present, and where, based on a special table, faulty addresses are remapped by a Built-In Repair Analysis (BIRA) unit [14];
- *Internal redundancy*, in which spare elements in the form of redundant rows and/or columns, are placed inside the memory alongside the normal columns and/or rows.

In some cases it is beneficial to merge both redundancy types and to include internal and external spares. For example, in a memory with multiple banks, each bank can have its own local internal spare memory, while at the same time global spare can co-exist.

In this work we consider *internal redundancy* only. Here, the mechanisms involved for row and column repair are quite different (see Figure 4.2). For *row replacement*, faulty row addresses detected by the BIST circuit are stored in special registers. Whenever the memory is accessed, the incoming address is first compared with those stored in the special registers to check if a defective row is to be accessed. If this is the case, the output of the comparator disables the row decoder and activates the spare wordline. For *column replacement*, in general, a column switching mechanism is present to isolate the faulty column and to forward data from non-defective cells [12]. In the example in the figure, all the bitlines located after the faulty bitline are shifted one spot to the left.

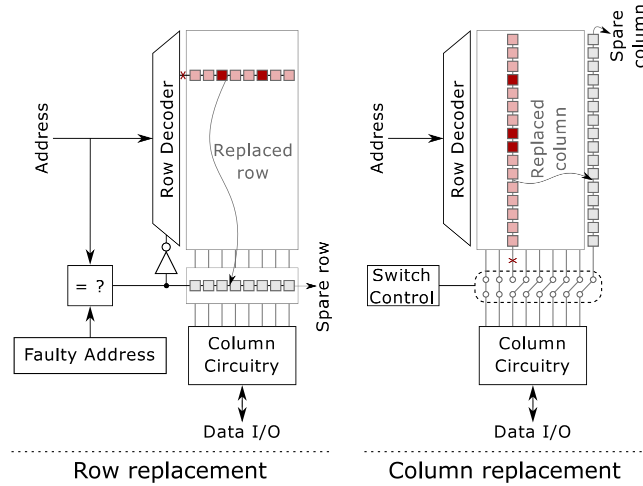


Figure 4.2: General 2D internal redundancy.

4.2.1 Previous Work in 3D Memory Repair

To create extra opportunities for memory repair various inter-die approaches have been proposed. The repair schemes usually consist of two major steps. In the first step, faulty cells are identified in a pre-bond test (i.e., a test prior to stacking). The second step allocates available spare resources, both local and inter-die spares, to maximize the compound yield. In [7] a Die-to-Die (D2D) stacking flow algorithm is presented which assumes that each die is beforehand locally repaired such that the number of available (not utilized for local repair) spares are made available as inputs for the global repair algorithm. This method for inter-die column replacement is suitable only for the particular case where arrays are simultaneously accessed with the same address. A similar D2D stacking approach is considered in [11] where the die stacking flow is modeled as a bipartite graph maximal matching problem. Several global D2D matching algorithms without local repair first are introduced and compared in [8]. An interesting approach is introduced in [9] where the authors propose to recycle irreparable dies (i.e., dies with arrays that are not repairable if only local spares are considered) in order to create good working memories. Their algorithm assumes Wafer-to-Wafer (W2W) stacking and the resulting memories comprise the good blocks from the bad dies.

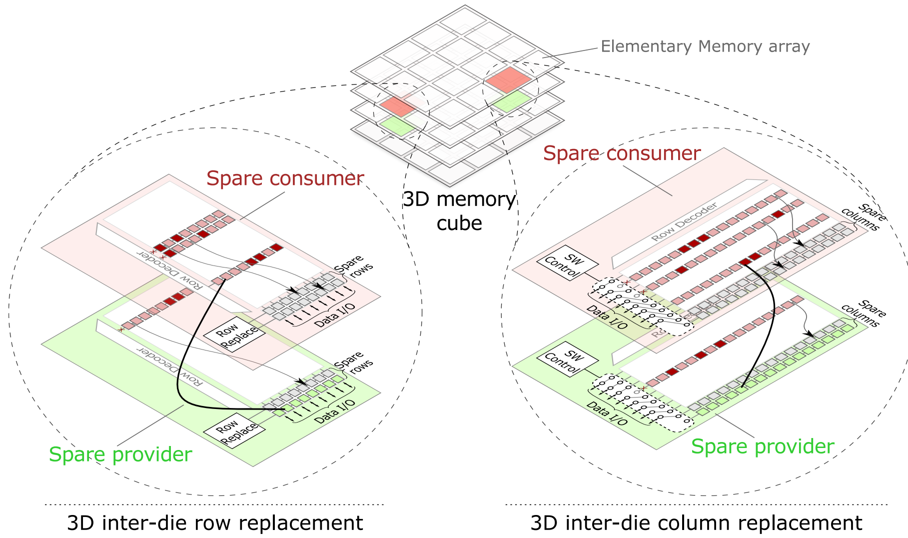


Figure 4.3: 3D inter-die memory repair - general idea.

4.3 3D Inter-die Memory Repair Architecture

The considered memory arrangement resembles a memory cube, as depicted in Figure 4.3. The cube employs 3D *array stacking* with the identical memory arrays being equipped with redundant rows and/or columns. In this organization, a situation may arise in which arrays with insufficient redundancy are in the vertical proximity of arrays that still have unutilized redundant elements. Supporting the replacement of faulty cells by using redundant resources from arrays from other dies, i.e., inter-die spare replacement, results in extended memory reparability rates [8]. This can be clearly observed in Figure 4.3, where on the left side, the top array has utilized all the available spare rows (two in this case) and still has one faulty row uncovered. However, the bottom array can provide the necessary spare row to replace the faulty one on the top array to make the memory defect free. Conversely, on the right side of the figure a similar scenario is depicted, but for inter-die column spare replacement.

We define the arrays that have available spare resources as *spare providers* and arrays which make use of the externally available spare resources as *spare consumers*. For the 3D memory repair to function correctly the consumer must be able to retrieve/store data from/on the provider in a transparent manner, i.e., the provider must be able to function normally, despite its spares being accessed by a neighboring die. In addition, it is important that the inter-die repair in-

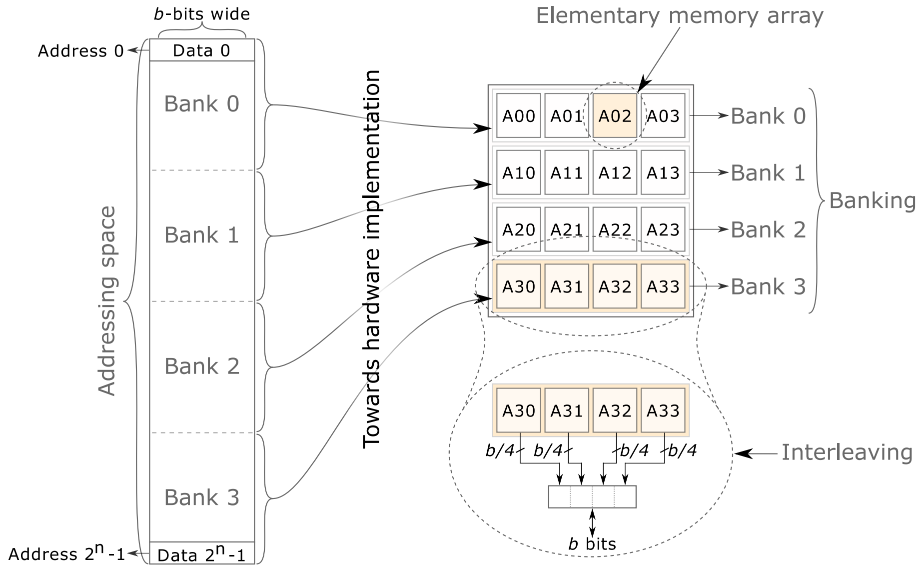


Figure 4.4: Memory partitioning.

frastructure does not disrupt the functionality of the memory cube when no repair takes place. Therefore, the required infrastructure that assures the memory repair mechanism is highly dependent on the exact internal structure of the memory arrays and it needs to function for a multitude of access patterns.

In order to balance area, delay, and power tradeoffs, a large memory is usually constructed in a hierarchical manner and is composed out of several banks, with each bank being further divided in several arrays. An example is presented in Figure 4.4 where the partitioning employs banking and interleaving. Each bank can be accessed either concurrently with independent addresses, or sequentially, where one bank is accessed while the rest remain idle. For interleaving, however, all the subarrays of a bank are concurrently accessed with the same address.

As the internal organization of the memory cube is defined at design time, a fixed memory partitioning implies three exclusive situations in which two memory arrays, a *provider-consumer* pair, can be accessed:

1. **Idle provider** - the two arrays are located in different banks that are never concurrently accessed; we use the term idle to denote that the two arrays are never accessed at the same time; from the *consumer*'s perspective this is equivalent with the *provider* being always idle;

2. **Busy provider with different access pattern** - the two arrays are located in different banks that are concurrently accessed with independent addresses;

3. **Busy provider with same access pattern** - the two arrays are part of the same bank with interleaving, therefore the accessing address is the same.

We add that, although our proposal is general and can in principle be applied to more than two adjacent dies, in this work we consider that inter-die replacement is performed between exclusive pairs of adjacent dies. The reasons behind this restriction are as follows: (i) the infrastructure overhead grows with the number of dies involved in the spare sharing process, and, (ii) two dies spare replacement is enough to sustain a satisfactory yield [8], since the die yield has a high value after repair. In the next section we introduce the infrastructure for the above identified *provider-consumer* repair schemes.

4.4 3D Inter-die Memory Repair Infrastructure

In this section we detail the inter-die repair schemes for each of the three *provider-consumer* pair scenarios introduced in Section 4.3 for row and column replacement.

4.4.1 Inter-die Row Replacement

Idle provider

Figure 4.5 depicts the situation for the case in which the *provider* is idle. On the *consumer* side, the local spare row is already allocated and another faulty row needs to be replaced remotely. A register is required to store its address (3DFR - 3D Fault Register), in a similar fashion as in the local replacement scheme. Furthermore, a comparator and several logic gates are introduced in the design to disable the local row decoder and to activate the spare wordline on the *provider* side whenever the incoming address is equal to the value stored in 3DFR. We propose to place the data TSVs after the column multiplexer. This requires the column address to be transfered through TSVs and the column decoder (CD) on the *provider* to be enabled. In this manner fewer TSVs are required when compared to the case when TSVs are placed for every bitline.

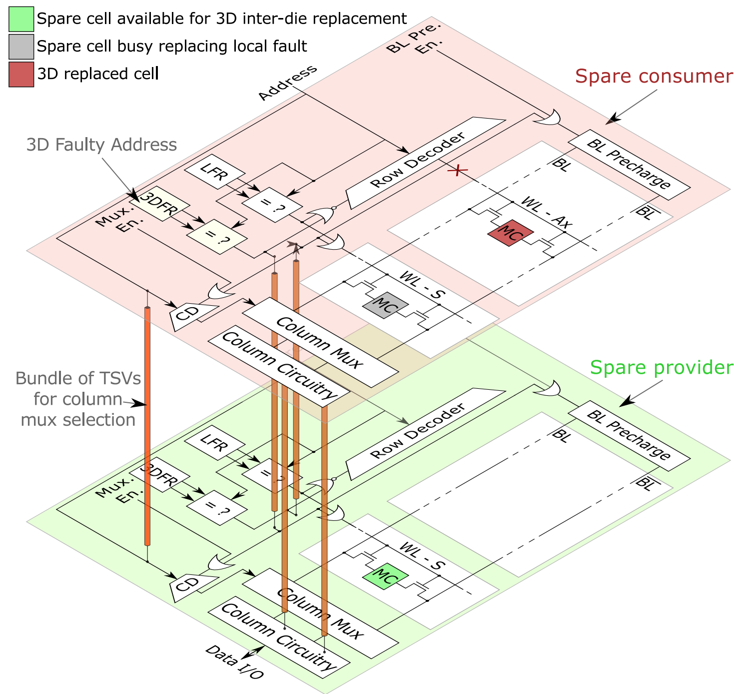


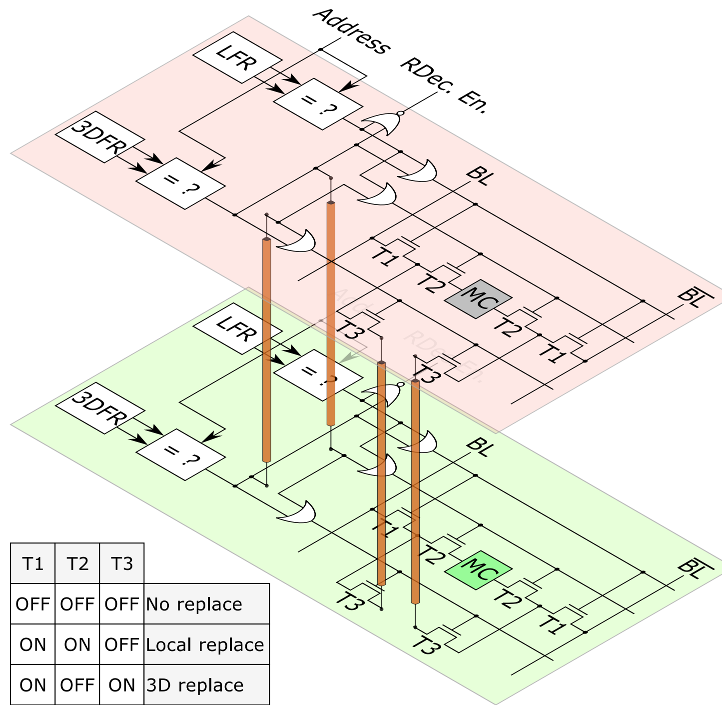
Figure 4.5: Inter-die row replacement infrastructure with idle *provider*.

Busy *provider* with different access pattern

When both *consumer* and *provider* can be accessed in parallel the constraints imposed to the inter-die memory repair interface are tighter, making the infrastructure more complex. In particular, when an inter-die replacement occurs, both *consumer* and *provider* need to use the *provider*'s bitlines, giving rise to a conflict. For this reason the data TSVs cannot be placed after the column muxes and extra transistors (denoted by T2 and T3) are required in every spare memory cell, as in Figure 4.6.

Busy *provider* with same access pattern

A particular case of *provider* and *consumer* parallel access arises when their address is the same (i.e., in an interleaving organization of memory banks, see Section 4.3). In contrast with the previous scheme, the infrastructure can be reduced in terms of logic. However, each spare cell still needs to be augmented with 4 extra transistors and 2 TSVs. Thus, even if we assume that future TSV



T1	T2	T3	
OFF	OFF	OFF	No replace
ON	ON	OFF	Local replace
ON	OFF	ON	3D replace

Figure 4.6: Inter-die row replacement infrastructure parallel *consumer* and *provider* access.

manufacturing process will be greatly improved to a negligible size, the cell area almost doubles.

We propose a scheme that alleviates the bitline conflict problem and makes use of fewer TSVs. For this, we take advantage of the concept of divided wordlines [15], a technique employed in SRAMs to reduce the number of active cells in a row. Our approach consists of separating the data paths for local access and inter-die access for the *provider*. In other words, we simplify the problem to the idle *provider* case. The mechanism to achieve this is to scramble the data in the spare row such that different bitlines are used on the *provider* array for accessing the local array data and the inter-die replacement data in the spare row. The scrambling pattern can be made persistent across different accesses since the same addresses are used to access the *consumer* and the *provider*. The inter-die replacement data are cross-mapped between exclusive pairs of two consecutive blocks, as in Figure 4.7. The data scrambler is replicated for every pair of consecutive blocks. The *provider's* 3DR bit controls the local data scrambling, to select the appropriate local wordline of the spare row, and

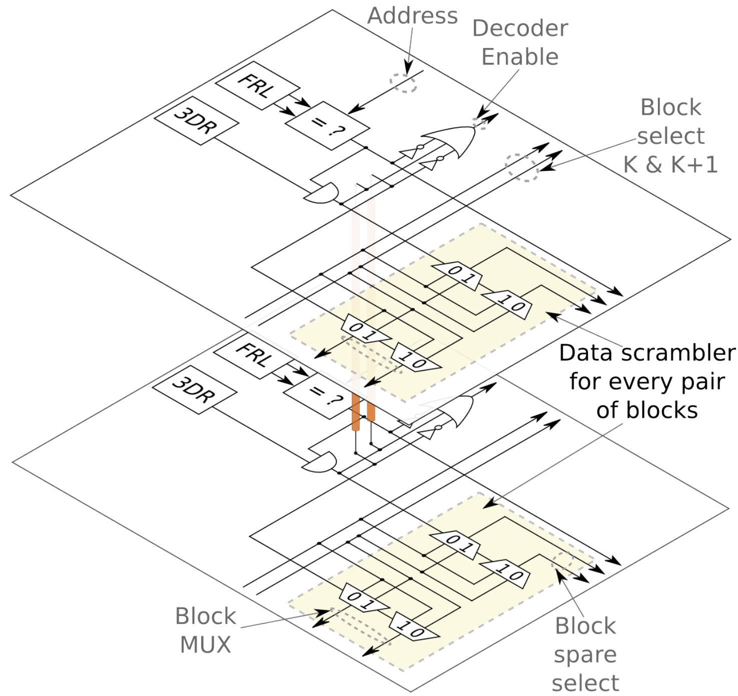


Figure 4.7: Inter-die row replacement with "busy *provider* identical access pattern", divided wordlines, detail.

at the same time de-scrambles the data on the *consumer* side by reversing the block mux selection.

4.4.2 Inter-die Column Replacement

The general infrastructure required for inter-die column replacement depicted in Figure 4.8 comprises all the cases introduced in Section 4.3. The common part for all the cases consists of the TSV pair utilized for bitline value transmission. They are enabled by the switching control block, which needs to be adapted to control also the inter-die replacement mechanism.

A special TSV is required for every wordline whenever the *provider* is busy accessing a different address, or when it is idle, in order to assert the required wordline for the *consumer*. When the *provider* is busy it is also mandatory to decouple the *provider's* wordline such that no bitline conflict arises because of multiple wordlines assertion. For brevity this action is not represented in Figure 4.8. For the case in which the *provider* is idle, the TSV required for the

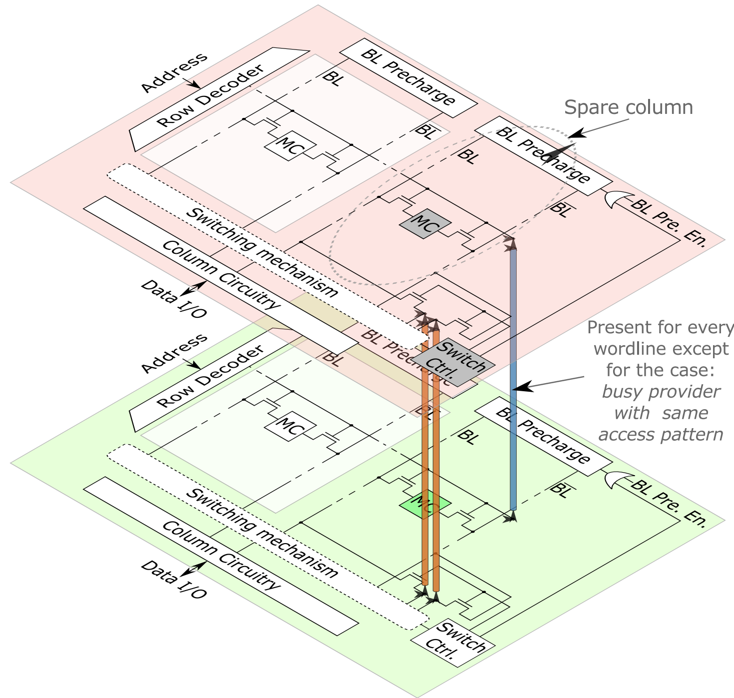


Figure 4.8: Inter-die row column replacement infrastructure.

wordline activation may be discarded if the *provider*'s row decoder is enabled. However, this requires the consumer's address to be driven onto the TSVs. Nevertheless, the gain is a significant TSV reduction.

The easiest and most convenient inter-die column replacement scheme in terms of TSV requirements is by far when the *consumer* and the *provider* are busy accessing the same address. Here, the same wordline is asserted in both arrays and no bitlines conflicts occur.

4.5 Discussion

In this section we discuss the overhead of the 3D inter-die memory repair schemes in terms of area and delay.

Area represents a sensitive issue in memory design and the memory cell is particulary the subject of severe scaling. SRAM bit cell has followed Moores's law, with an area shrinking rate of about 1/2 for every generation, reaching 0.081 μm^2 for the 22 nm technology [16]. This rate is expected to last even in

Table 4.1: TSV requirements for proposed schemes

Memory access scenarios	Number of TSVs
Row replacement	
Idle *provider*	$2 \times spares + 2 \times dw + cd_bits$
Busy *provider* with different address	$spares \times (2 + 2 \times columns)$
Busy *provider* with same address	$spares \times (2 + 2 \times columns)$
Busy *provider* with DWL	$2 \times spares + 4 \times dw \times blocks$
Column replacement	
Idle *provider*	$2 \times spares + log_2(rows)$
Busy *provider* with different address	$2 \times spares + rows$
Busy *provider* with same address	$2 \times spares$

* *dw* = data width (data I/O); *cd_bits* = column decoder input bits.

the realm of post-CMOS devices [17]. TSVs dimensions are predicted to scale down too, but not that steep as SRAM bit cells. The predictions from [18] suggest a gradually decreasing trend with a shrinking ratio of about $1/4$ for every 3 years, reaching a minimum diameter of 0.8 μm and a pitch of 1.6 μm by 2018. Nowadays manufactured TSVs have a diameter bewteen 3 and 10 μm and a pitch of about 10 μm [19; 20; 21]. From their large size it is clear that TSVs represent the major contributor to the 3D memory repair area overhead.

Table 4.1 presents the TSV requirements for all the scenarios introduced in Section 4.3. The scenarios that have the least number of TSVs are "idle *provider*" for row redundancy and "busy *provider* with same access" for column redundancy. All the other scenarios require a large number of TSVs that make them absolutely impractical. Even for the row redundancy with the "idle *provider*" scenario the practicality is problematic. Figure 4.9 depicts the area of one redundant row and its required TSVs. The redundant row area is drawn for different technology nodes using memory widths varying between 128 and 2048 bits. The TSV area is independent of the technology node and is calculated for a TSV pitch between 0.5 and 3.5 μm and a memory data width of 32 bits. Given that, inter-die redundancy may be profitable only if the redundant row area is smaller than the TSV area. Figure 4.9 clearly suggests that for a large TSV pitch inter-die redundancy becomes impractical. However, this can be improved if there are more spares that use the same data TSVs.

It is interesting to find the required TSV pitch for which inter-die row replacement becomes advantageous. Table 4.2 shows the TSV pitch required such that the TSV overhead is equal to a single spare row for 16 nm 6T-SRAM cell.

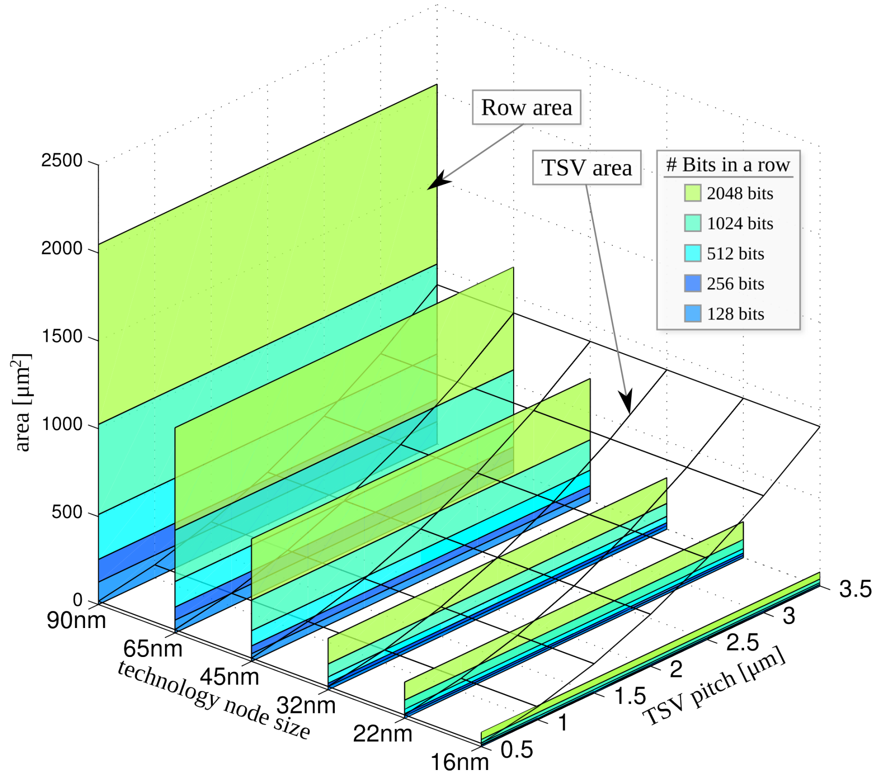


Figure 4.9: TSV area overhead vs. row area for 32-bit data I/O.

For example, in case the column width is 512 bits and the data output width is 32 bits, the TSV pitch must be at most 534 nm. For the worst case considered configuration, with a column width of 512 bits and data output width of 64 bits, TSV pitch needs even to scale further down to 388 nm. Therefore, current TSV sizes, which are in the order of 3 μm, need to be shrinked severely for inter-die redundancy to be beneficial for a wide range of memory configurations.

The access time (T_{N2D}) for a normal memory read operation (Eq. (4.1)) is determined by: address decoding (T_{dec}), wordline generation (T_{WL}), bitlines discharge (T_{BL}), column multiplexing (T_{mux}), and data sensing (T_{SA}). If row redundancy is present and the redundant row is accessed the access time changes to T_{R2D} (Eq. (4.2)), because the access goes through the comparator (T_{cmp}) instead of the decoder. For 3D row redundancy, extra time is required to transfer data to the *consumer* through the TSVs (T_{TSV}), resulting in T_{R3D} (Eq. (4.3)). The time overhead for 3D row redundancy (D_{OR}) can be computed as in Eq. (4.4). For 3D inter-die column redundancy, the access time increases

Table 4.2: Required TSV pitch [μm] for 16nm 6T-SRAM cell

Columns	Data width				
	4	8	16	32	64
128	0.576	0.476	0.367	0.270	0.195
256	0.789	0.658	0.512	0.380	0.275
512	1.083	0.911	0.715	0.534	0.387

as in Eq. (4.6). Thus, there is always a delay overhead (D_{OC}) due to TSV propagation and switching time.

$$T_{N2D} = T_{dec} + T_{WL} + T_{BL} + T_{mux} + T_{SA} \tag{4.1}$$
$$T_{R2D} = T_{cmp} + T_{WL} + T_{BL} + T_{mux} + T_{SA} \tag{4.2}$$
$$T_{R3D} = T_{R2D} + 2 \times T_{TSV} \tag{4.3}$$
$$D_{OR} = \frac{max(T_{N2D}, T_{R3D}) - max(T_{N2D}, T_{R2D})}{max(T_{N2D}, T_{R2D})} \tag{4.4}$$
$$T_{C2D} = T_{N2D} + T_{switching} \tag{4.5}$$
$$T_{C3D} = T_{C2D} + T_{switching} \tag{4.6}$$
$$D_{OC} = \frac{T_{switching} + T_{TSV}}{T_{C2D}} \tag{4.7}$$

As the delay of a TSV is in the order of 20 ps [22], we expect the following to hold true: $T_{R2D} < T_{R3D} < T_{N2D}$. Therefore, no delay penalty for row repair is expected. For column repair however, the following inequation holds: $T_{N2D} < T_{C2D} < T_{C3D}$. The overhead is determined by the delay of switching muxes and a TSV ($T_{switching} + T_{TSV}$) which is expected to be minimal.

4.6 Conclusion

In this chapter, we presented a study of inter-die repair schemes for TSV based 3D-SICs, i.e., of using repair in the vertical dimension by accessing spares of neighboring stacked dies, in addition with the traditional column and row repair. We provided an overview of general repair schemes and subsequently, proposed a memory framework for inter-die redundancy based on a *provider-consumer* pair scheme. We proposed different infrastructures for different memory organizations (based on their access mechanisms), in which the infrastructure handles the data transfer from *provider* to *consumer* in a transpar-

ent mode without interfering with the *provider*'s operation. Our analysis suggests that for state-of-the-art TSV dimensions inter-die column-based repair schemes could result in yield improvements at a reasonable area overhead. For row repair, however, most memory configurations require TSV dimensions to scale down at least with one order of magnitude to be efficiently utilized in 3D memory systems.

Note. The content of this chapter is based on the following paper:

M. Lefter, G.R. Voicu, M. Taouil, M. Enachescu, S. Hamdioui, S.D. Cotofana, **Is TSV-based 3D Integration Suitable for Inter-die Memory Repair?**, *Design, Automation & Test in Europe (DATE)*, Grenoble, France, March 2013.

Bibliography

[1] P. Garrou, *Handbook of 3D integration: Technology and Applications of 3D integrated circuits*. Weinheim: Wiley-VCH, 2008.

[2] K. Puttaswamy and G. H. Loh, “3D-Integrated SRAM components for high-performance microprocessors,” *TC*, 2009.

[3] “ITRS - System Drivers,” tech. rep., 2011.

[4] S. Rusu, M. Sachdev, C. Svensson, and B. Nauta, “Trends and challenges in VLSI technology scaling towards 100nm,” in *VLSI Design / ASPDAC*, 2002.

[5] S. R. Nassif, “The light at the end of the CMOS tunnel,” *ASAP*, 2010.

[6] R. Anigundi, H. Sun, J.-Q. Lu, K. Rose, and T. Zhang, “Architecture design exploration of three-dimensional (3D) integrated DRAM,” in *ISQED*, 2009.

[7] C. Chou, Y. Huang, and J. Li, “Yield-enhancement techniques for 3D random access memories,” in *VLSI-DAT*, 2010.

[8] L. Jiang, R. Ye, and Q. Xu, “Yield enhancement for 3D-stacked memory by redundancy sharing across dies,” in *ICCAD*, 2010.

[9] Y.-F. Chou, D.-M. Kwai, and C.-W. Wu, “Yield enhancement by bad-die recycling and stacking with though-silicon vias,” *TVLSI*, 2011.

[10] C.-W. Wu, S.-K. Lu, and J.-F. Li, “On test and repair of 3D random access memory,” in *ASPDAC*, 2012.

[11] S. Lu, T. Chang, and H. Hsu, “Yield enhancement techniques for 3-dimensional random access memories,” *Microelectronics Reliability*, 2012. Cited by 0001.

[12] M. Horiguchi and K. Itoh, *Nanoscale Memory Repair*.

[13] P. Ohler, S. Hellebrand, and H.-J. Wunderlich, “An integrated built-in test and repair approach for memories with 2D redundancy,” ETS, 2007.

[14] N. Axelos, K. Pekmestzi, and D. Gizopoulos, “Efficient memory repair using cache-based redundancy,” *TVLSI*, 2011.

[15] M. Yoshimoto, K. Anami, H. Shinohara, T. Yoshihara, H. Takagi, S. Nagao, S. Kayano, and T. Nakano, “A divided word-line structure in the static RAM and its application to a 64K full CMOS RAM,” *JSSC*, 1983.

[16] K. Smith, A. Wang, and L. Fujino, “Through the looking glass: Trend tracking for ISSCC 2012,” *M-JSSC*, 2012.

[17] H. Iwai, “Roadmap for 22nm and beyond,” *Microelectronic Eng.*, 2009.

[18] “ITRS - Interconect,” tech. rep., 2011.

[19] C. L. Yu, C. H. Chang, H. Y. Wang, J. H. Chang, L. H. Huang, C. W. Kuo, S. P. Tai, S. Y. Hou, W. L. Lin, E. B. Liao, *et al.*, “TSV process optimization for reduced device impact on 28nm CMOS,” in *TVLSI*, 2011.

[20] G. Katti, A. Mercha, J. Van Olmen, C. Huyghebaert, A. Jourdain, M. Stucchi, M. Rakowski, I. Debusschere, P. Soussan, W. Dehaene, *et al.*, “3D stacked ICs using cu TSVs and die to wafer hybrid collective bonding,” in *IEDM*, 2009.

[21] H. Chaabouni, M. Rousseau, P. Leduc, A. Farcy, R. El Farhane, A. Thuaire, G. Haury, A. Valentian, G. Billiot, M. Assous, *et al.*, “Investigation on TSV impact on 65nm CMOS devices and circuits,” in *IEDM*, 2010.

[22] D. H. Kim and S. K. Lim, “Through-silicon-via-aware delay and power prediction model for buffered interconnects in 3D ICs,” in *SLIP*, 2010.

5

Low Cost Multi-Error Correction for 3D Polyhedral Memories

In this chapter we propose a novel error correction scheme/architecture specially tailored for polyhedral memories which: (i) allows for the formation of long codewords without interfering with the memory architecture/addressing mode/data granularity and (ii) make use of codecs located on a dedicated tier of the 3D memory stack. For a transparent error correction process we propose an online memory scrubbing policy that performs the error detection and correction decoupled from the normal memory operation. To evaluate our proposal we consider as a case study a 4-die 4-MB polyhedral memory and simulate various data width codes implementations. The simulations indicate that our proposal outperforms state of the art single error correction schemes in terms of error correction capability, being able to diminish the Word Error Rates (WER) by many orders of magnitude, e.g., WER from 10^{-10} to 10^{-21} are achieved for bit error probabilities between 10^{-4} and 10^{-6}, while requiring less redundancy overhead. The scrubbing mechanism hides the codec latency and provides up to 10% and 25% write and read latency reductions, respectively. In addition, by relocating the encoders/decoders from the memory dies to a dedicated one a 13% footprint reduction is obtained.

5.1 Introduction

Increased integration factor and technology shrinking make Integrated Circuits (ICs) more prone to different defect types during the manufacturing process [1] and to in field degradations [2]. With memory cell size reduction, Multi-Bit Upsets (MBUs) become much more frequent since a particle hitting a memory array creates disturbances in multiple physically adjacent cells [3; 4; 5; 6]. In [7], a maximum MBU bit multiplicity of over 100 bits is predicted for 32 and 22nm SRAM generations, thus making traditional Single-Error Correction (SEC) ECCs with column interleaving [8] not any-longer sufficient [9] to mitigate the large amount of MBUs. In view of this, alternative approaches, like ECCs with Multi-Error Correction (MEC) capabilities become of high interest.

Three dimensional stacked ICs (3D-SICs) based on Through-Silicon-Via (TSV) interconnects [10] rely on an emerging technology which further boost the trends of increasing transistor density [11] and performance, since it enables smaller footprints, high bandwidth low latency interconnection and heterogeneous integration, while facilitating dependable computing [12]. While most of the 3D memory designs just follow a certain folding strategy, the 3D polyhedral memory architecture proposed in [13] brings a different fresh view into the field. It consists of multiple identical memory banks stacked on top of each other, while TSVs bundles distributed across the entire memory footprint traverse all the stacked dies to enable an enriched memory access set not achievable in planar counterparts.

In this chapter we introduce a novel error correction scheme/architecture tailored for polyhedral memories. The main idea behind our approach consists in performing MEC ECC encoding/decoding on larger data widths (e.g., 512 to 4096 bits) such that a better error correction capability is obtained with the same or even lower redundancy than the one required by state of the art 64 data bit SEC schemes. For this we exploit the polyhedral memories organization, which inherently fast and customizable wide-I/O vertical access mechanisms allow for the long codeword creation with data and check bits from multiple mats. Furthermore, we propose to make use of codecs located on a dedicated tier of the 3D memory stack, since polyhedral memories allow for a smooth transfer of those long codewords to/from the dedicated error correction tier. In order to make the error correction process transparent to the memory users, e.g., processing cores, we propose an online memory scrubbing policy that performs the error detection and correction decoupled from the normal memory operation. The proposed approach create the premises for a large design space from where the most appropriate scheme can be chosen by balancing the

desired error correction capability while fulfilling different overhead budgets in terms of memory footprint, latency, and energy.

We evaluated our proposal by considering as a case study a 4-die 4-MB polyhedral memory protected by the proposed MEC mechanism with various data width codes implementations. The simulation experiments indicate that our proposal outperform state of the art schemes in terms of error correction capability, by significantly diminishing the Word Error Rates (WER), e.g., WER from 10^{-10} to 10^{-21} are achieved for bit error probabilities between 10^{-4} and 10^{-6}. Additionally, the scrubbing mechanism hides the codec latency and provides up to 10% and 25% write and read latency reductions, respectively. Furthermore, by relocating the encoders/decoders from the memory dies to a dedicated one a 13% footprint reduction is obtained and parallel energy effective scrubbing can be enabled, which results in even larger WER reductions.

The outline of the chapter is the following. In Section 5.2 we describe our MEC error correction proposed approach. In Section 5.3 we evaluate the implications of our proposals and perform a comparison with state of the art memory error correction approaches. Section 5.4 concludes the chapter.

5.2 ECC for Polyhedral Memories

In this section we detail our proposal for a Multi-Error Correction (MEC) ECC mechanism specially tailored for polyhedral memories [13]. The main goal is to allow for the formation of long data width codewords, case in which BCH codes are more effective, while not changing the normal memory addressing granularity, which in most of the current architectures is 64 bit. Based on the encoders/decoders (codecs) placement we detail next two scenarios: (i) *On Die Internal ECC* and (ii) *Dedicated Die ECC*. We note that the two scenarios can be employed independently or synergistic.

5.2.1 On Die Internal ECC

In this scenario the codecs could be physically located on each die, i.e., coplanar with the memory dies, as depicted in Figure 5.1. The h-bit data I/O of each sub-bank is split in j groups of h/j bits on which on the fly encoding/decoding is applied on memory write/read accesses. The total number of required codecs is equal with the number of groups (j) multiplied with the number of sub-banks in a die.

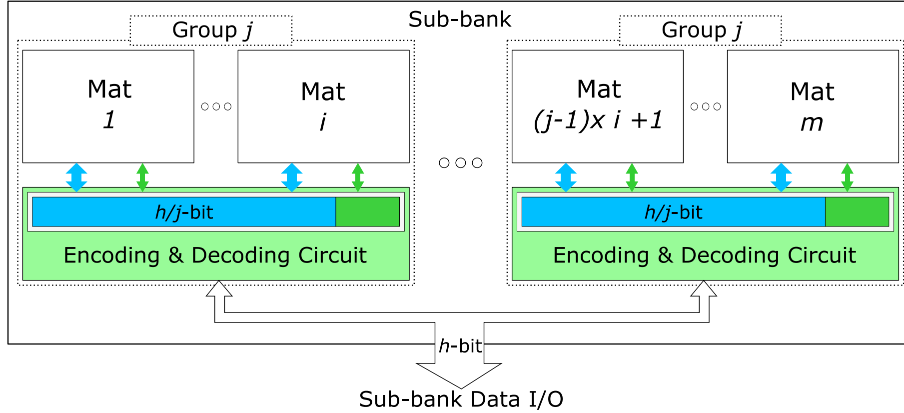


Figure 5.1: *On die internal ECC* for polyhedral memories.

5.2.2 Dedicated Die ECC

As detailed in [13], polyhedral memories allow for vertical data access at different granularities: various sub-banks could be accessed in parallel on different dies, with the largest amount of vertically accessed data being obtained when all the TSVs are utilized. To better exploit the polyhedral memories rich access set we propose the *Dedicated Die ECC* scenario, depicted in Figure 5.2. Specifically, we propose to: (i) form extended codewords by combining data and check bits from multiple mats, but which could now be part of different sub-banks, as opposed to the previous scenario which was restricted to combining data only from mats in the same sub-bank, (ii) augment the 3D memory stack with an extra die dedicated to the execution of the encoding and decoding actions, (iii) employ the TSVs to transfer the extended codewords on/from the dedicated error correction die, and (iv) rely on an online programmable scrubbing based memory maintenance policy, which performs the error detection and correction activities with as little interference as possible with data read/write requests coming from the SoC computation cores. Based on the required error protection level, the online memory maintenance policy operates in one of the following modes: **scrubbing only**, **on the fly write with scrubbing**, and **on the fly write and read**.

Memory maintenance operates mainly in the **scrubbing only** mode, in which, at certain time periods (scrubbing intervals), the entire memory is scanned (in a time period which we call scrubbing time) and eventual errors are corrected. The scrubbing interval and the scrubbing time depend on: (i) the employed MEC code performance (ECC encoding/decoding latency), (ii) the memory

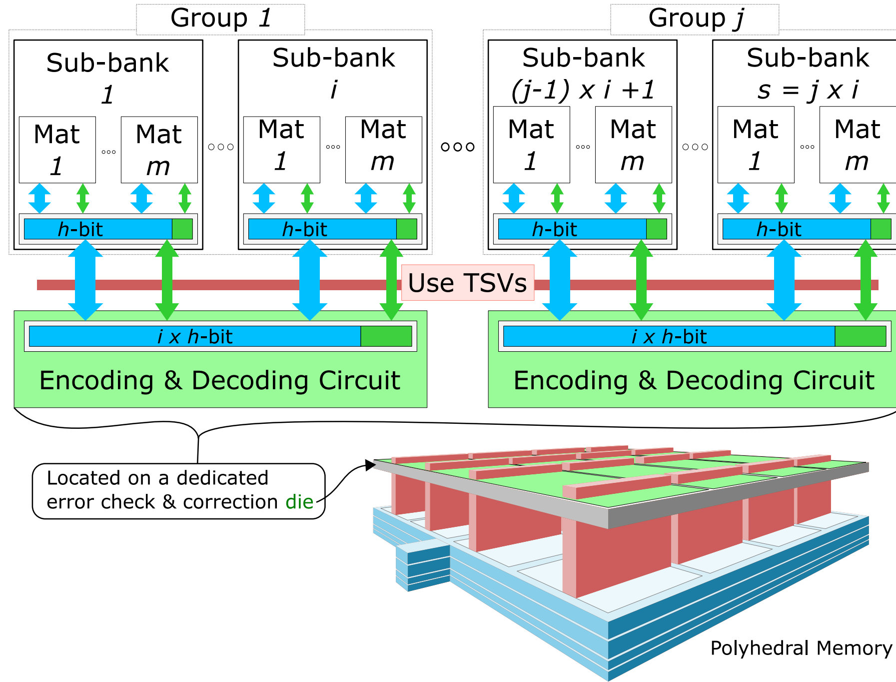


Figure 5.2: MEC-based ECC for polyhedral memories.

latency, (iii) the number of MEC codecs available on the error correction die, (iv) the eventual memory conflicts that may occur, (v) the overall memory capacity, and, (vi) the bit error occurrence rate, which depends on aging and environmental aggressions.

The scrubbing maintenance operates in a similar way the DRAM refresh does and in case of conflicts its read/write accesses have lower priority than normal read/write memory accesses issued by the computation cores. We note that, different from the DRAM refresh, memory scrubbing requires additional steps related to the codeword formation and transfer to/from the codec die. Moreover, coding and/or decoding it is much more time consuming than a write-back thus by implication maintenance related memory accesses are less frequent than the computational related ones. Consequently, in order to optimize both scrubbing intervals and scrubbing times, the scrubbing controller can take advantage of: (i) the polyhedral memory reach access mode set and (ii) the low maintenance related accesses occurrence, to dynamically adapt its access schedule such that memory conflicts are minimised if not completely avoided. In this way memory error correction activities are executed in back-

ground and no decoding/coding actions are part of normal memory read/write accesses. This has the main advantage that the memory access time is practically unaffected by its augmentation with MEC ECC capabilities. However, if the memory operates in highly aggressive environmental conditions, or aging effects become predominant, resulting in high error rate, it is possible that the scrubbing process cannot keep the pace in cleaning the induced bit-flips. In this case faulty data might be read by the application(s) running on the platform embedding the ECC protected memory which may result in unpredictable application behaviour. Note that not all the bit-flips are malicious as they might be rewritten before being read again. Such situations can be detected by the Operating System (OS), which can decide to change the ECC modus operandi to an on de fly coding/encoding mode.

Note that even if from the memory user point of view a data write seems coding free, it launches an encoding process too. This is because a write at address A invalidates all the check bits of codeword C (to which the data item at address A belongs). Consequently, the codeword C data bits need to be transferred via TSVs on the dedicated error correction die in order to be re-encoded, after which the newly calculated check bits are written back into the memory (data bits could be also rewritten for more protection). Invalidated codewords are handled by the scrubbing process and depending on the bit error rate we could place the invalidated codeword in the front of the scrubbing queue or to take no further action as there is a large chance that another write may occur within the same codeword in the close future. Note that the write invalidation policy may have an impact on the error correction capability. Eventual bit errors which may appear between the invalidation and the actual check bits update become indiscoverable, as when they are read in relation with the codeword re-encoding they will be considered valid.

If the aging and environmental conditions ask for a better protection, the **on the fly write with scrubbing** mode could be employed, in which the eventual write invalidation errors described above are avoided. When a word write is requested at address A the following actions have to take place:

1. Load the codeword C (to which address A belongs) on the error correction die via the afferent TSV bundle.

2. Decode C and correct possible errors if any.

3. Replace the value at A with the new to be written value.

4. Re-encode the codeword C.

5. Store C back to the memory via the same TSV bundle.

While the **on the fly write with scrubbing** operation mode eliminates the write invalidation problem, errors could still propagate in the following manner: eventual bit errors which may appear between two scrubbing scans could still propagate when read operations on those locations are performed. To avoid such situations the **on the fly write and read mode** can be employed. In this operation mode when a read request at address A (which is part of codeword C) is issued, the following actions have to take place:

1. Load via the afferent TSV bundle the entire codeword C on the codec die.
2. Decode C and correct possible errors if any.
3. Send the (corrected) data value stored at A to the computation core that issued the request.
4. In case errors were found at Step 2:
   - Re-encode the codeword C.
   - Store C back via the same TSV bundle.

We note that from the SoC (memory user) point of view, which is the one that really matters, the memory maintenance is 100% successful when it can keep pace with the error formation rate such that data read generated by the running application(s) never return corrupted data. This does not mean that all memory locations should be error free, which is quite normal as memory locations not currently read by the application do not influence its behavior thus they may contain erroneous bits. The error protection operation modes can be dynamically switched by the application by monitoring the scrubbing effectiveness. In this way the protection level is adjusted to the environmental aggression level and SoC aging status. We note however that the execution of codec activities during read/write operations as done when "on the fly" modes are activated has detrimental impact on SoC performance in terms of throughput, latency, and energy consumption.

The 3D memory is constructed by stacking identical memory dies which dimensions determine the IC footprint. In view of this the error correction die can be as large as the memory dies, thus for state of the art memory capacities it might be able to accommodate more than one MEC codec, case in which the memory maintenance process can be parallelized resulting in a higher error

resilience. Finally, it is worth noticing that if memory accesses have a larger granularity, the MEC (re-)coding overhead is diminishing, and the read/write complexity can be reduced by, e.g., write buffers, caching.

To explore the potential of our proposal, we evaluate in the next section the error correction capability of our approach and compare it with the one of state of the art approaches.

5.3 Performance Evaluation

In this section we evaluate our MEC ECC proposal for polyhedral memories. First, we assess the reliability gains induced by the utilization of extended codewords instead of the traditional SEC ECCs. Second, we evaluate the implications of our proposal in terms of memory footprint, access time, and energy consumption. In our simulations we consider as a case study a 4 die 4-MB polyhedral memory with a 512-bit wide horizontal interface and a 4096-bit wide vertical interface. On each die there are 8 sub-banks (with an 512-bit I/O width), each being composed of 8 mats (with an 64-bit I/O width).

5.3.1 Error Correction Capability

To evaluate our proposal performance in terms of reliability for different redundancy ratios, we simulated the following protections mechanisms:

1. State of the art 64-bit SEC ECC (**SEC_CONV**).
2. Binary BCH codes ([14], [15]) operating on up to 4096-bit, denoted as **D/C/E**, with D, C, E as the number of data bits, check bits, and correctable errors, respectively.

To simulate memory fault occurrences, we made use of a Binary Symmetric Channel (BSC) model with crossover probabilities (α), i.e., the probability that a memory bit is being flipped, from 10^{-6} up to 10^{-2}. The performance of all the considered designs in terms of Word Error Rate (WER) assessed by means of Monte Carlo simulations is graphically depicted in Figure 5.3. The traditional SEC_CONV ECC protected memory, plotted with dark blue solid lines and circle markers, is considered as a reference, thus in the figure we plotted: (i) polyhedral memory implementations with higher WER (e.g., 128/8/1) and/or with higher redundancy ratio (check bits overhead, e.g., 128/24/3) when

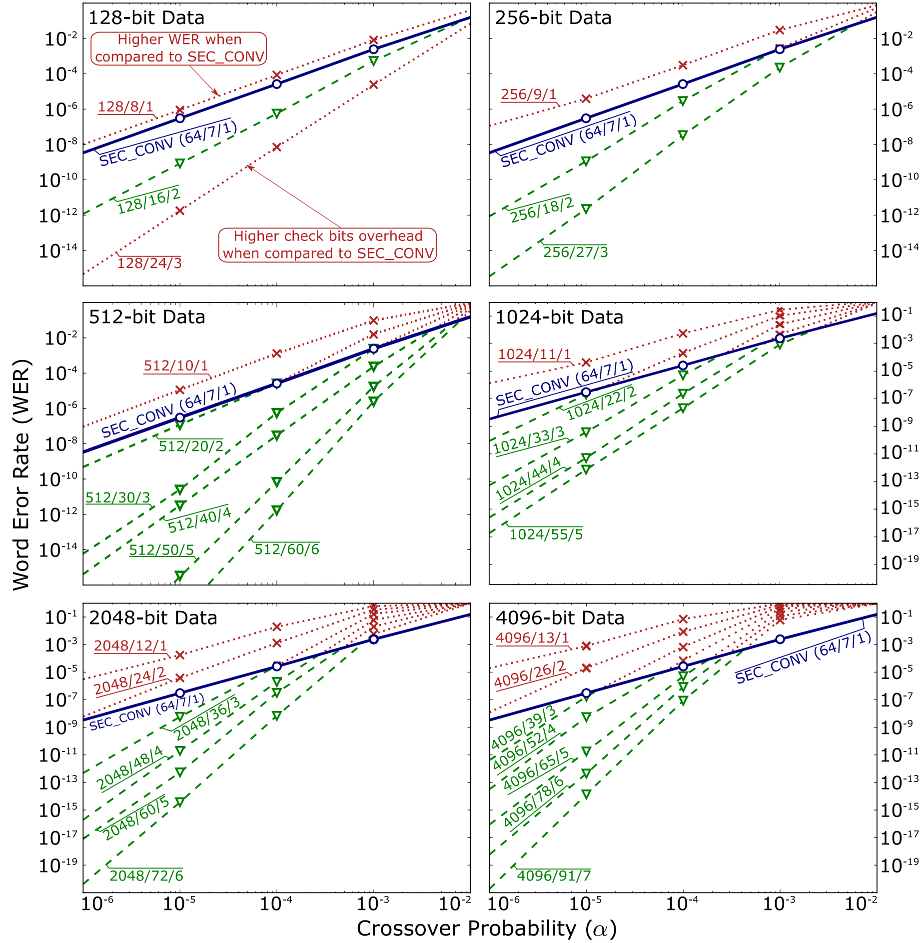


Figure 5.3: Word error rate vs bit flip probability.

compared with SEC_CONV with dotted red lines and x markers, while, (ii) polyhedral memories with smaller WER and lower redundancy with green dashed lines and V markers.

From Figure 5.3 one can deduce that there is no "panacea universalis" in place, but most of the D/C/E polyhedral memory configurations outperform the state of the art in terms of WER at the expense of lower redundancy requirements. In particular, we can observe the following main situations:

1. D/C/E configurations with small data width and not enough redundancy, i.e., 128/8/1 and 256/9/1, which provide a higher than SEC_CONV

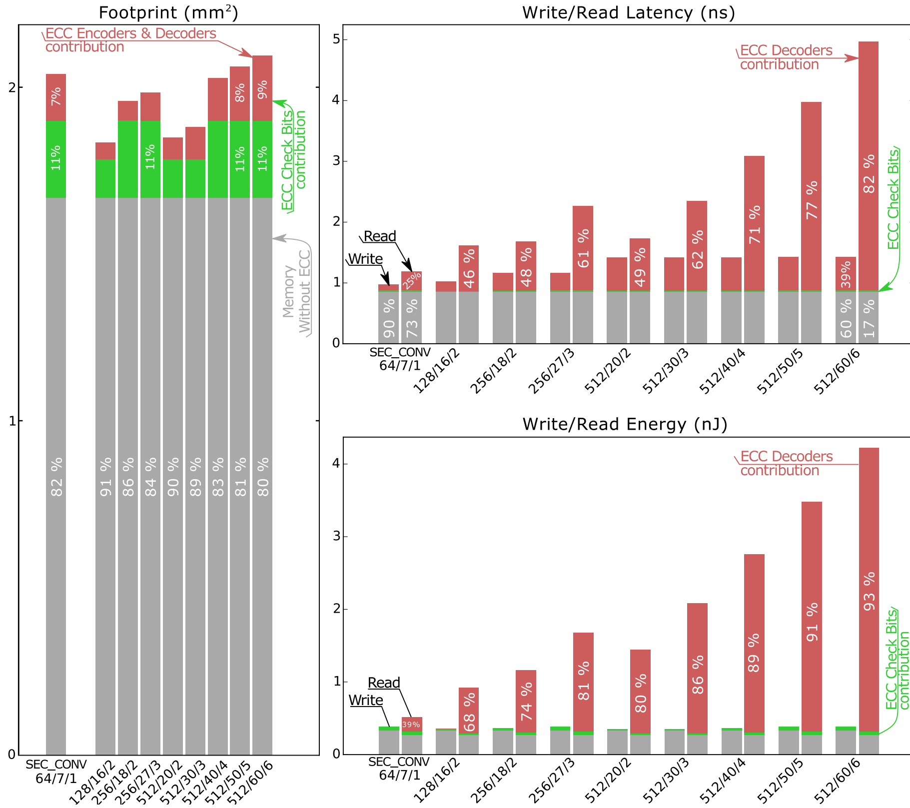


Figure 5.4: On die internal ECC evaluation.

WER for the entire considered α domain (not of practical interest).

2. D/C/E configurations with small-medium data width and low redundancy, i.e., 128/16/2, 256/27/3, 512/50/5, which substantially outperform SEC_CONV in terms of WER for the entire considered α domain while also diminishing the required redundancy by, e.g., 20% in the $256/27/3$ case.

3. D/C/E configurations with medium-large data width and extremely low redundancy, i.e., $1024/44/4$, $2048/60/5$, $4096/78/6$, which outperform by many orders of magnitude SEC_CONV in terms of WER for $\alpha < 10^{-4}$ while also diminishing the required redundancy by, e.g., 4.2x in the $2048/60/5$ case.

Figure 5.3 clearly demonstrates that our approach creates a large design space one can explore in his/her quest for the most appropriate D/C/E configuration

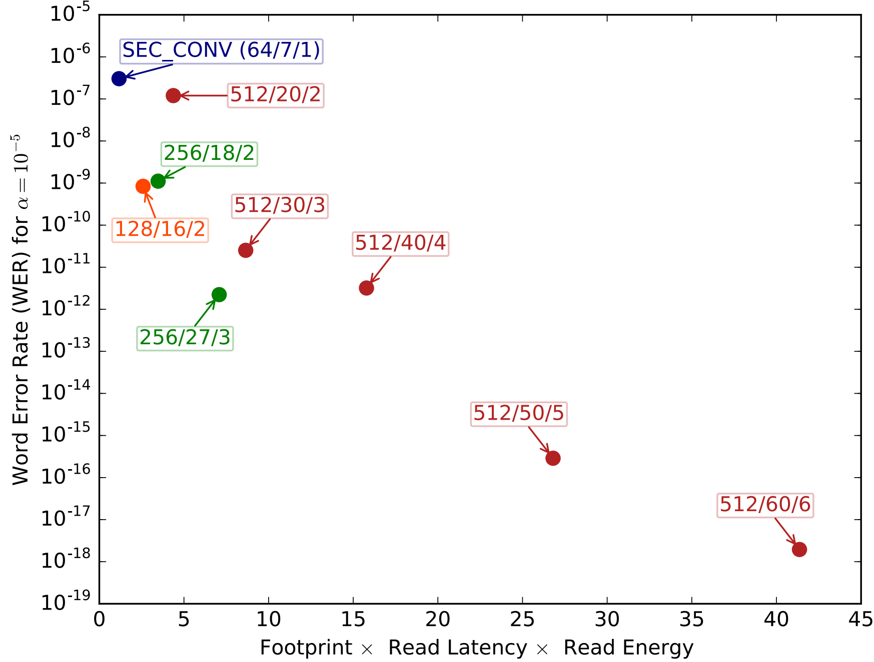


Figure 5.5: On die internal trade offs.

corresponding to the expected memory operation conditions, i.e., the α range, and targeted maximum acceptable WER value. For example, if the operating conditions require $\alpha \leq 10^{-5}$ and WER $\leq 10^{-12}$ any of the following configurations can be utilized: 512/40/4, 1024/44/4, 2048/60/5, and 4096/78/6. The actual choice can be done based on criteria like the lowest redundancy ratio but other aspects related to the memory architecture and utilization may also play a role in the best candidate selection. We mention that the above results can be regarded as a lower bound and that we expect even lower WER for BCH protected memories if a more realistic fault injection model, dedicated for MEC, is employed.

5.3.2 On Die Internal ECC Overhead

While in the previous section we demonstrated that our proposal can substantially outperform state of the art ECC approaches with similar or less redundancy, it is of interest to evaluate the actual implications of the considered MEC ECC schemes in terms of die footprint, memory access time, and energy consumption. To this end we considered a 22nm CMOS technology and esti-

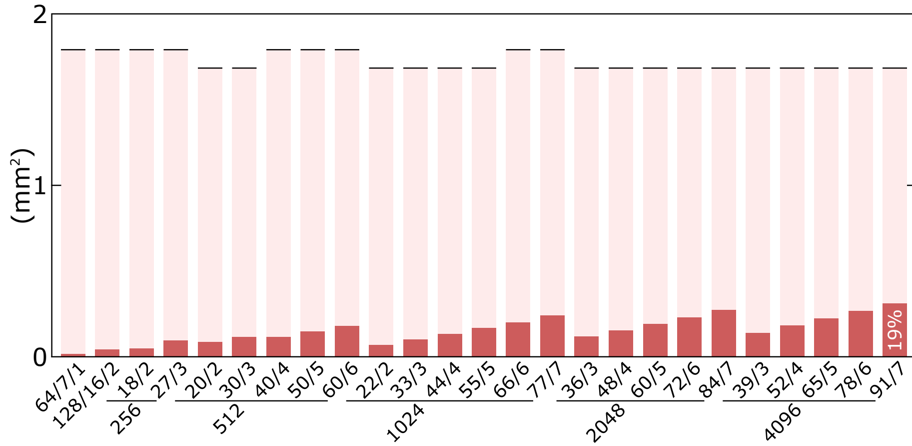


Figure 5.6: Dedicated die ECC - footprint comparison.

mated the footprint, latency, and energy of the utilized BCH encoders/decoders by using formulas from [16; 17; 18]. Additionally we obtained the ECC augmented memory dies footprint, latency, and energy values with a modified Cacti 6 [19] simulator utilizing the approach from [13]. We have evaluated only the MEC BCH protected memories which, in the previous reliability subsection, proved better in terms of error correction capability. The results of our evaluations are depicted in Figure 5.4.

One can conclude from the footprint graph (Figure 5.4 left) that: (i) the combined ECC area contribution (including check bits and codecs) is smaller than 20% for all of the considered ECC implementations, and (ii) the codecs overhead is always smaller than the check bits overhead. When comparing the implementations, the SEC_ CONV implementation requires one of the biggest footprint, being matched only by the largest 512-bit codecs considered (512/50/5 and 512/60/6). We note that the 128/16/2 implementation provides a 10% footprint reduction when compared with SEC_CONV while substantially outperforming it in terms of WER (see Figure 5.3).

In terms of latency (Figure 5.4 center), the ECC contributions are increasing when larger data width codecs are employed, reaching a maximum of 40% for write operations, while for read operations the ascending trend is much more pronounced, reaching a maximum of 83%. The ECC contribution to the memory access time is almost entirely determined by the codecs and all of the **D/C/E** MEC ECC implementations exhibit an increased latencies when compared to SEC_CONV.

In terms of energy consumption (Figure 5.4 right) a significant increase is observed for read operations for the ECC schemes spanning from 39% for SEC_CONV and going up to 93% for larger data width MEC codecs. For write operations the encoder contributions are insignificant (less than 3%) for all the implementations.

In order to get a better insight into the possible trade-offs, we have plotted on the bottom left of Figure 5.4 the WER (for a constant crossover probability $\alpha = 10^{-5}$) of all the considered approaches against the compound footprint $\times$ read latency $\times$ energy metric. One can observe in the figure that there is no absolute winer. The SEC_CONV implementation is the best in terms of the compound metric, while it is the worse in terms of error correction capability: it provides a WER which is with less then 2 orders of magnitude smaller than the considered α. On the other hand, the 512/60/6 implementation is the best in terms of error correction capability (it provides a WER which is with 13 orders of magnitude smaller than the considered α) but it is the worse in terms of the footprint $\times$ read latency $\times$ energy metric. More balanced options are, e.g., 128/16/2 or 256/27/3 implementations, and given that at design time one knows the expected α range and the maximum acceptable WER value one can make use of such a plot in order to identify the most appropriate MEC ECC organization.

5.3.3 Dedicated Die External ECC Analysis

Since the *Dedicated Die External ECC* approach has a special error correction die, in Figure 5.6 we plotted the polyhedral memory footprint (left side of the figure) and the codec area on the error correction die (right side) for all the evaluated **D/C/E** schemes. Notice that the ECC die has a substantial amount of available area as even the largest codec is not utilizing more than 20% of its real-estate. This means that the remaining area could be either employed by extra codecs, to improve the error correction capability, or by other SoC resources. We also note that all of the *Dedicated Die External ECC* approaches provide a footprint reduction of 10% or 13% (depending on the required check bits) when compared with the SEC_CONV *On Die Internal ECC*, mostly because the codecs have been relocated on a dedicated die.

As mentioned in Section 5.2.2, there are various operation modes for the *Dedicated Die External ECC* approach. Here, we limit our discussion to the scrubbing operation mode - the preferred modus operandi as in this use case no ECC related latency penalty per memory access is induced, irrespectively of the employed MEC ECC. This translates into write and read latency reduc-

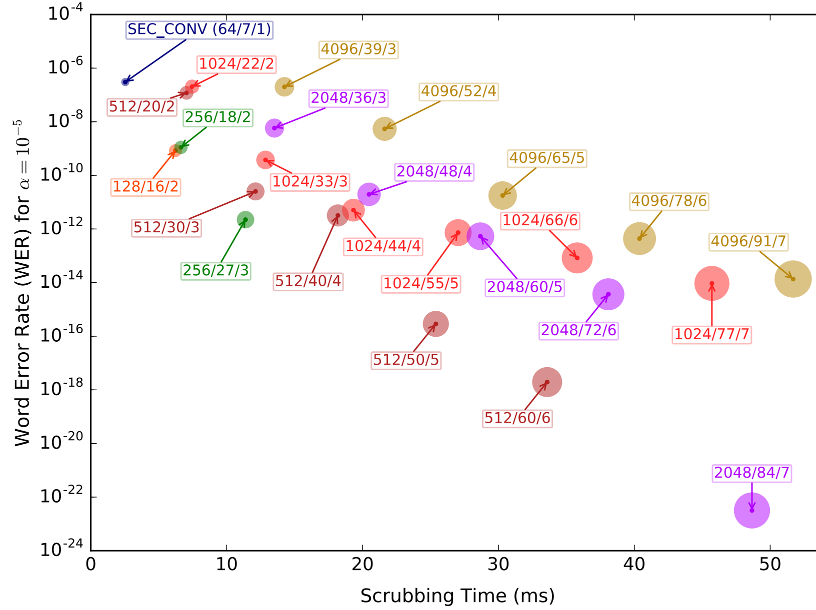


Figure 5.7: WER vs. ideal scrubbing time.

tions of 10% and 25%, respectively, when compared to the SEC_CONV *On Die Internal ECC* scheme.

An important metric for the *Dedicated Die External ECC* approach is the scrubbing time, which is essential to be as short as possible, in order for the error correction controller to cope with high error rates. In Figure 5.7 we assume a bit error probability $\alpha = 10^{-5}$ and present the relation between the scrubbing time and the obtained codecs WER. We evaluated a scrubbing time lower bound as we assumed that no memory access conflicts occurred between the normal SoC memory accesses and scrubbing related TSV traffic. In practice such conflicts may occur but most can be solved by an adaptive scheduling policy. Figure 5.7 shows that the SEC_CONV implementation has the fastest scrubbing time but it is very weak in terms of error correction capability. In contrast, the 2048/84/7/6 implementation is the best in terms of error correction capability but it is the worst in terms of scrubbing time. Many other possibilities exist between the two extremes, one of the most balanced option being 512/50/5.

Regarding energy, again there are no direct penalties incurred per memory accesses but the *Dedicated Die External ECC* scrubbing mechanism consumes

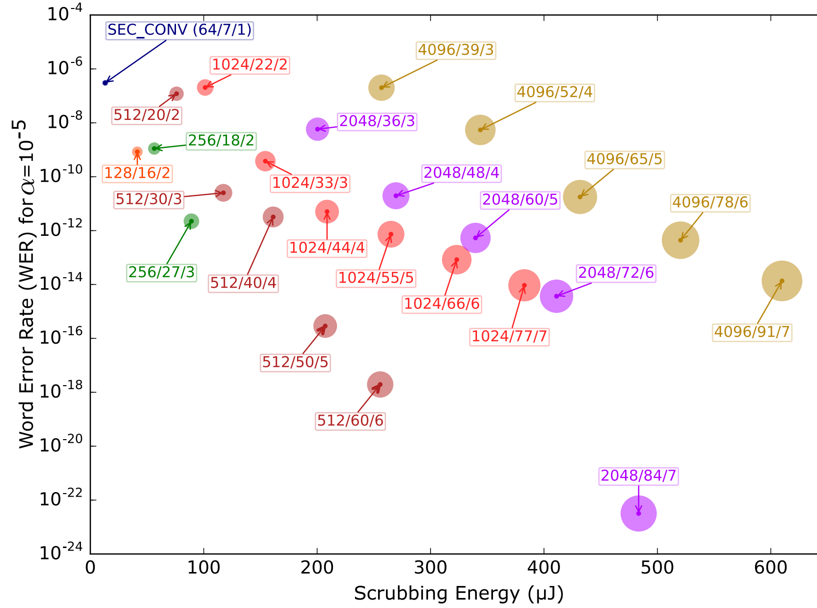


Figure 5.8: WER vs. ideal scrubbing energy.

energy whenever it performs a memory scan. One can observe in Figure 5.8 that the SEC_CONV implementation performs best in terms of energy (it consumes the least amount of energy), while it is the worse in terms of error correction capability. At the same time the 2048/84/7/6 implementation is the best in terms of error correction capability but it is the worse in terms consumed energy. Again, one of the most balanced options is 512/50/5.

We note however that if more encoders/decoders are employed on the ECC die (as there is available room for that), the scrubbing time can be reduced. For example, if seven 2048/84/7 codecs are employed (such that the entire ECC die is utilized), the ideal scrubbing time is reduced to almost the same value as the one corresponding to SEC_CONV, and 2048/84/7 becomes the most balanced option from the WER vs. ideal scrubbing time perspective.

5.4 Conclusion

In this chapter we introduced a novel error correction mechanism for 3D wide-I/O polyhedral memories. The main idea behind our approach was to create the premises for applying ECC on larger data widths such that MEC can be

performed with the same or even lower redundancy than the one required by state of the art 64 data bit SEC schemes. In addition, we proposed an online memory scrubbing policy that can perform the error detection and correction decoupled from the normal memory operation. We evaluated our proposal by considering a 4-die 4-MB polyhedral memory as a case study and simulated various data width codes implementations. The simulations indicated that our proposal outperforms state of the art schemes in terms of error correction capability, being able to diminish the WERs by many orders of magnitude, while requiring less redundancy overhead. Moreover, by relocating the codecs on a specialized die in the 3D memory stack we managed to hide the codec latency and provided 10% and 25% write and read latency reductions, respectively. At the same time a 13% footprint reduction was obtained.

Note. The content of this chapter is based on the following paper:

M. Lefter, T. Marconi, G.R. Voicu, S.D. Cotofana, **Low Cost Multi-Error Correction for 3D Polyhedral Memories**, *IEEE/ACM International Symposium on Nanoscale Architectures (NANOARCH)*, Newport, USA, July 2017.

Bibliography

[1] S. Nassif, "The light at the end of the cmos tunnel," in *ASAP*, July 2010.

[2] S. Rusu, M. Sachdev, C. Svensson, and B. Nauta, "T3: Trends and challenges in vlsi technology scaling towards 100nm," in *ASP-DAC*, 2002.

[3] K. Mohr and L. Clark, "Delay and area efficient first-level cache soft error detection and correction," in *Computer Design, 2006. ICCD 2006. International Conference on*, pp. 88–92, Oct 2006.

[4] N. Seifert, P. Slankard, M. Kirsch, B. Narasimham, V. Zia, C. Brookreson, A. Vo, S. Mitra, B. Gill, and J. Maiz, "Radiation-induced soft error rates of advanced cmos bulk devices," in *Reliability Physics Symposium Proceedings, 2006. 44th Annual., IEEE International*, pp. 217–225, March 2006.

[5] G. Georgakos, P. Huber, M. Ostermayr, E. Amirante, and F. Ruckerbauer, "Investigation of increased multi-bit failure rate due to neutron induced seu in advanced embedded srams," in *VLSI Circuits, 2007 IEEE Symposium on*, pp. 80–81, June 2007.

[6] A. Dixit and A. Wood, "The impact of new technology on soft error rates," in *Reliability Physics Symposium (IRPS), 2011 IEEE International*, pp. 5B.4.1–5B.4.7, April 2011.

[7] E. Ibe, H. Taniguchi, Y. Yahagi, K. Shimbo, and T. Toba, "Impact of scaling on neutron-induced soft error in srams from a 250 nm to a 22 nm design rule," *Electron Devices, IEEE Transactions on*, vol. 57, pp. 1527–1538, July 2010.

[8] T. Suzuki, Y. Yamagami, I. Hatanaka, A. Shibayama, H. Akamatsu, and H. Yamauchi, "0.3 to 1.5v embedded sram with device-fluctuation-tolerant access-control and cosmic-ray-immune hidden-ecc scheme," in *ISSCC*, 2005.

[9] R. Naseer and J. Draper, "Parallel double error correcting code design to mitigate multi-bit upsets in srams," in *ESSCIRC*, 2008.

[10] P. Garrou, C. Bower, and P. Ramm, *3D Integration: Technology and Applications*. Wiley-VCH, 2008.

[11] B. Prince, *Vertical 3D Memory Technologies* . Wiley, 2014.

[12] S. Safiruddin, D. Borodin, M. Lefter, G. Voicu, and S. D. Cotofana, "Is 3d integration the way to future dependable computing platforms?," in *2012 13th International Conference on Optimization of Electrical and Electronic Equipment (OPTIM)*, pp. 1233–1242, May 2012.

[13] M. Lefter, G. Voicu, and S. Dan Cotofana, "A shared polyhedral cache for 3d wide-i/o multi-core computing platforms," in *ISCAS*, 2015.

[14] A. Hocquenghem, "Codes Correcteurs d'Erreurs," *Chiffres (Paris)*, 1959.

[15] R. C. Bose and D. K. Ray-Chaudhuri, "On a class of error correcting binary group codes," *Information and Control*, 1960.

[16] X. Zhang and K. Parhi, “High-speed architectures for parallel long bch encoders,” *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 13, pp. 872–877, July 2005.

[17] D. Strukov, “The area and latency tradeoffs of binary bit-parallel bch decoders for prospective nanoelectronic memories,” in *Signals, Systems and Computers, 2006. ACSSC ’06. Fortieth Asilomar Conference on*, pp. 1183–1187, Oct 2006.

[18] Y. Sankarasubramaniam, I. Akyildiz, and S. McLaughlin, “Energy efficiency based packet size optimization in wireless sensor networks,” in *IEEE International Workshop on Sensor Network Protocols and Applications*, pp. 1–8, 2003.

[19] N. Muralimanohar, R. Balasubramonian, and N. P. Jouppi, “CACTI 6.0: A tool to model large caches,” tech. rep., HP Laboratories, 2009.

6

LDPC-Based Adaptive Multi-Error Correction for 3D Memories

In this chapter we introduce a novel error resilient memory architecture potentially applicable to a large range of memory technologies. In contrast with state of the art memory error correction schemes, which rely on (extended Hamming) Error Correcting Codes (ECC), we make use of Low Density Parity Check (LDPC) codes due to their close to the Shannon performance limit error correction capabilities. To allow for a cost-effective implementation we build our approach on top of a 3D memory organization which inherently fast and customizable wide-I/O vertical access allows for a smooth transfer of the required LDPC long code-words to/from an error correction dedicated die. To make the error correction process transparent to the memory users, e.g., processing cores, we propose an online memory scrubbing policy that performs the LDPC-based error detection and correction decoupled from the normal memory operation. For evaluation purposes we consider 3D memories protected by the proposed LDPC mechanism with various data width codes implementations. Simulation results indicate that our proposal clearly outperforms state of the art ECC schemes with fault tolerance improvements by a $4710\times$ factor being obtained when compared to extended Hamming ECC. Furthermore, we evaluate instances of the proposed memory concept equipped with different LDPC codecs implemented on a commercial 40nm low-power CMOS technology and evaluate them on actual memory traces in terms of error correction capability, area, latency, and energy. Our results indicate that the LDPC protected memories offer substantially improved error correction capabilities, when compared to state of the art extended Hamming ECC, being able to assure clean runs for memory error rates $\alpha < 3 \times 10^{-2}$, which demonstrate that our proposal can potentially successfully protect system on a chip memory systems even in very harsh environmental conditions.

6.1 Introduction

Technology shrinking and increased integration factor allow, on one hand, for continuous Integrated Circuits (ICs) performance improvements. On the other hand, ICs are more prone to different defect types during the manufacturing process [1] and to in field degradations [2]. Multi-Bit Upsets (MBUs) become much more frequent [3; 4] with a maximum MBU bit multiplicity of over 100 bits being predicted for 32 and 22nm SRAM generations [5]. Thus, traditional Single-Error Correction (SEC) ECCs with column interleaving [6] cannot anylonger mitigate this large amount of MBUs [7] and powerful but cost effective techniques to detect and correct multiple memory errors are becoming crucial for future SoC related developments [8].

Three dimensional stacked ICs (3D-SICs) based on Through-Silicon-Via (TSV) interconnects [9] further boost increased transistor density and performance [10; 11; 12] while facilitating dependable computing [12; 13; 14]. Various 3D memory designs have been proposed ever since the technology was introduced [15; 16; 17; 18]. In particular for polyhedral memories [16], TSVs bundles are distributed across the entire memory footprint. This enables a bandwidth amount and a wide interconnect width not achievable in planar counterparts, which opens new avenues for memory error correction [19].

In this chapter we propose a novel memory error correction mechanism which takes advantage of the 3D memories flexible, powerful, and wide data access capabilities. Our approach relies on performing Low Density Parity Check (LDPC) encoding/decoding [20] on large codewords, which can be quickly transferred over the 3D memory TSVs to a dedicated die, on which the actual error correction and detection is performed. An important component of the mechanism consists of an online memory scrubbing policy, which enables transparent error detection and correction. In case the scrubbing process cannot keep the pace in cleaning the memory bit-flips induced by harsh environmental conditions or aging effects, faulty data might be read by the processing units, which may result in unpredictable application behavior. To handle this issue, we propose an adaptation mechanism controlled by the Operating System (OS) to enable *write with on the fly encoding* and *read with on the fly decoding*, which increase the memory access time but allow for memory integrity preservations even in extremely aggressive environments.

We evaluated our proposal by considering as a case study 3D memories protected by the proposed LDPC mechanism with various data width codes implementations. The simulation experiments indicate that our proposal outper-

forms state of the art extended Hamming ECC schemes in terms of error correction capability, being able to tolerate crossover probability rates (α) up to 8×10^{-2} for a targeted Word Error Rate (WER) of 10^{-6}. This translates into fault tolerance improvements by a $4710\times$ factor when compared to extended Hamming ECC. To assess the practical implications of our proposal in a realistic scenario, we implement instances of the proposed memory concept equipped with different LDPC codecs as Application-Specific Integrated Circuit (ASIC) on a commercial 40nm low-power CMOS technology and evaluate them on actual memory traces in terms of error correction capability, area, latency, and energy. Our results indicate that the LDPC based protection memories offer substantially improved error correction capabilities, when compared to state of the art extended Hamming ECC, being able to assure correct application execution for bit flip error rates $\alpha < 3 \times 10^{-2}$ which demonstrate that our proposal can potentially successfully protect memory systems even in very harsh environmental conditions.

The outline of the chapter is the following. In Section 6.2 we describe our LDPC-based error correction proposed approach. In Section 6.4 we evaluate the implications of our proposal and perform a comparison with state of the art memory error correction approaches. Section 6.5 concludes the chapter.

6.2 LDPC-based Error Correction Proposal

The proposed LDPC error protection mechanism depicted in Figure 6.1 consists in: (i) forming extended codewords by combining data and check bits from multiple memory sources, (ii) augmenting the 3D memory stack with extra logic dedicated to the LDPC encoding and decoding execution, (iii) employing TSVs to transfer the extended codewords on/from the dedicated Error Correction (EC) die, and (iv) utilizing an online memory scrubbing policy able to perform memory maintenance without interfering with data requests issued by the System on Chip (SoC) computation cores.

Memory maintenance is performed by means of a **scrubbing** procedure, i.e., at certain time periods (*scrubbing intervals*), the entire memory is scanned (in a time period which we call *scrubbing time*) and eventual errors are corrected. See Figure 6.2 for timing details. We mention that Memory Read (MR) and Memory Write (MW) operations handle memory I/O width data while Srubbing Read (SR) and Scrubbing Write (SW) operate on large data width values upper bounded by the number of TSVs in the 3D memory. The scrubbing maintenance operates in a transparent manner, i.e., in case of conflicts SR/SW

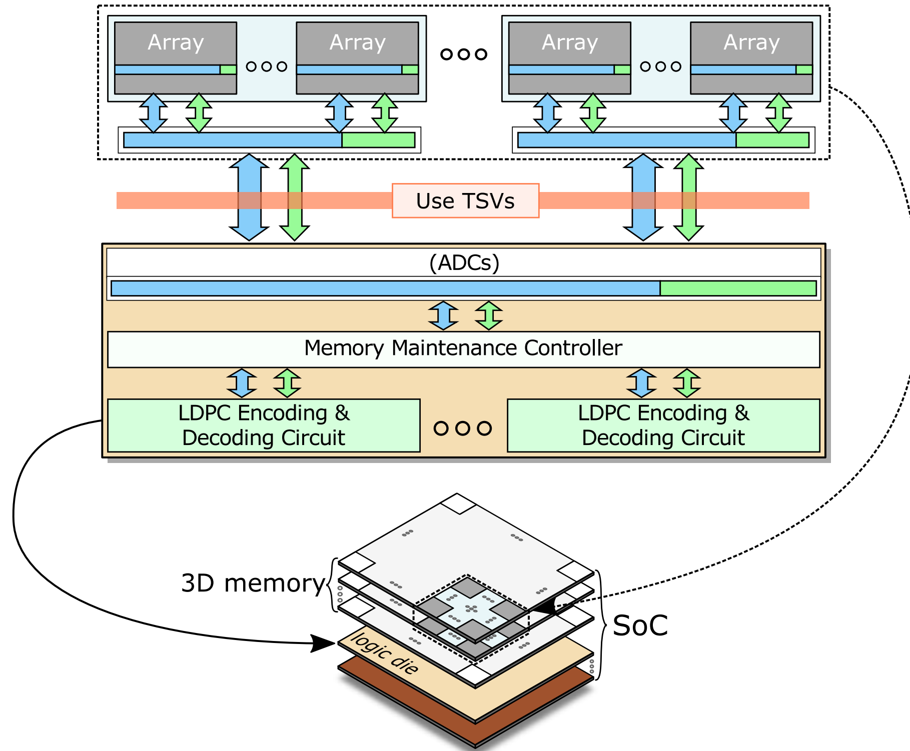


Figure 6.1: LDPC-based error correction for 3D memories.

accesses have lower priority than normal MR/MW accesses. Due to the 3D memory organization [16] multiple accesses can be served in parallel as long as no arrays and/or TSV conflicts are incurred.

The scrubbing procedure steps are depicted in Figure 6.3. At memory system start-up a scrubbing initialization process is performed (*step 0*). This may include, e.g., (i) memory initialization, (ii) LDPC codec allocation and instantiation as the codec dedicated tier may contain more than one LDPC codec, (iii) scrubbing start address(es) allocation. In *step 1* the LDPC codeword comprising parts located at the to be currently scrubbed address are brought on the codec tier. This step may require several SR accesses in various arrays which can be performed serially or in parallel. When all the comprising codeword parts are loaded, they can be provided as input to the LDPC decoder such that *step 2*, i.e., the actual decoding, can proceed. In case the LDPC decoder fails to identify a valid codeword at *step 2* an exception is raised which treatment is up to the application (OS) policy in place. A successful decoding triggers

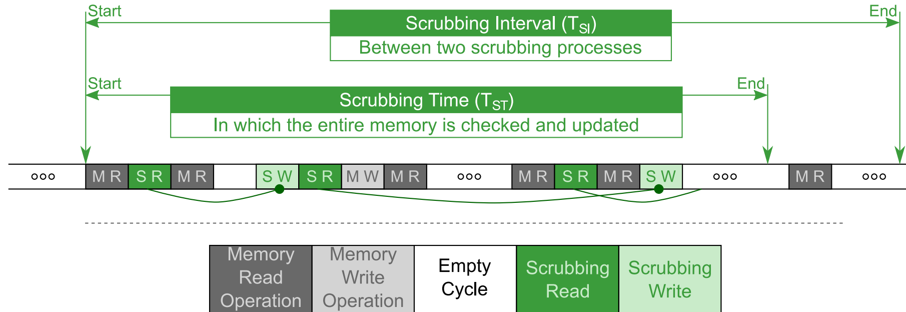


Figure 6.2: Memory scrubbing timeline.

step 3, in which possible errors are identified by comparing the decoder input codeword with the decoder output codeword. Eventual discovered errors are reported and a codeword update process is initiated (*step 4*). The codeword update process is similar to *step 1*, with the difference that now SW accesses are performed. In *step 5* the next to be scrubbed address is computed and if this is the initial to be scrubbed address, the scrubbing iteration ends and *step 6* in which the scrubbing process is idle is entered, otherwise, the flow restarts from *step 1*. The scrubbing time depends on: (i) employed LDPC codec performance, (ii) memory latency (*steps 1,4*), (iii) LDPC codecs number (*step 2*), (iv) memory array and TSV conflicts (*steps 1,4*), (v) overall memory capacity, and, (vi) bit-error rate, which depends on aging and environmental aggression profile. It can be noticed that memory scrubbing requires additional steps related to the codeword formation and transfer to/from the codec die. Moreover memory scrubbing coding activity is much more time consuming than a write-back thus by implication maintenance related memory accesses are less frequent than the computation related ones. Consequently, in order to diminish the scrubbing time, the scrubbing controller can take advantage of: (i) the 3D memory rich access mode set, and (ii) the low maintenance related accesses occurrence, to dynamically adapt its access schedule such that memory conflicts are minimized, if not completely avoided.

Even though from the user point of view the proposed memory system data write seems coding free, a launch of an encoding process (and sometimes also of a decoding process) is required when a write at address A invalidates all the codeword C check bits to which the data item at address A belongs. Consequently, the codeword C data bits need to be transferred via TSVs on the dedicated EC die in order to be re-encoded and written back into the memory. Depending on the actual bit error rate we may decide to place the invalidated message in the front of the scrubbing queue or to take no further action as

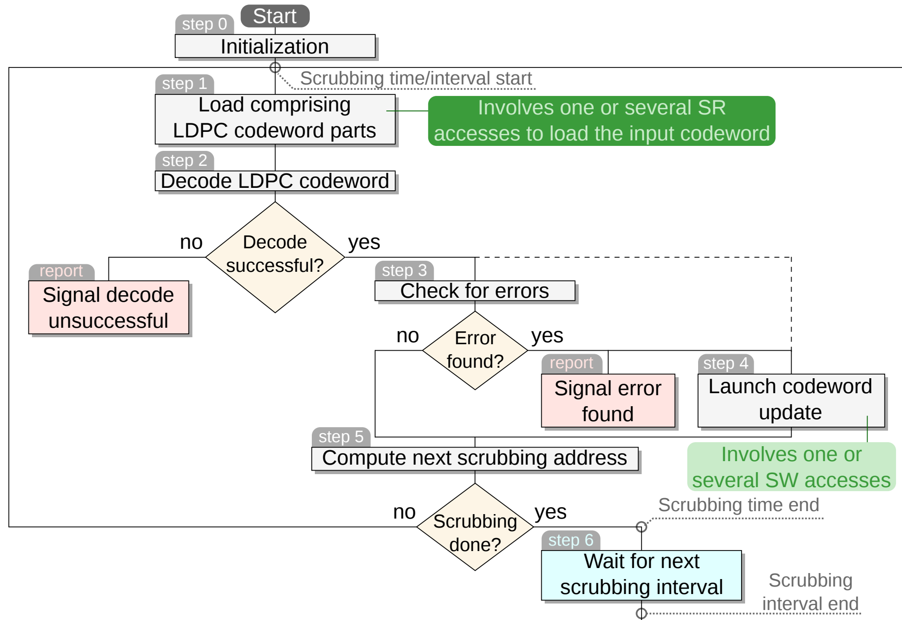


Figure 6.3: Memory maintenance scrubbing flow.

there is a large chance that another write may occur within the same message in the close future. Nevertheless, a read from an invalidated message should immediately trigger the reconstruction of the check bits. We refer to this write policy as *write invalidation* and we note that it may have an impact on the EC efficiency. If aging and environmental conditions ask for a better protection, a *write with on the fly encoding* operation should be employed. Thus, when a word write is requested at address A a series of actions have to take place. First, the codeword C (to which address A belongs) has to be loaded on the EC die. Next, C is decoded, the possible errors if any are corrected and the value at address A is replaced with the new to be written value and codeword C is re-encoded. Finally, C is stored back into the memory. Regardless of the write policy, errors could still affect memory integrity as bit-flips occurring between two scrubbing scans could propagate when read operations on those locations are performed. To avoid such situations the *read with on the fly* operation can be employed, such that a read request at address A (which is part of codeword C) comprises the following actions: (i) load the codeword C on the codec die, decode it, and correct possible errors, if any (ii) send the data value stored at A to the computation core that issued the request, and (iii) re-encode the codeword C, if errors were identified, and store it back to the memory.

If memory operates in highly aggressive environmental conditions or aging effects become predominant resulting in high error rate it is possible that the scrubbing process cannot keep the pace in cleaning the induced bit-flips. Such situations can be detected by the Operating System (OS), which can decide to change the ECC modus operandi and to enable *write with on the fly encoding* and *read with on the fly* operations. We note however that the execution of codec activities during read/write operations required when on the fly modes are activated might have a detrimental impact on SoC performance in terms of throughput, latency, and energy consumption. Since 3D memories are constructed by stacking identical memory dies which dimensions determine the IC footprint, the EC die might be able to accommodate several LDPC codecs. Hence, the memory maintenance process can be parallelized resulting in a higher error resilience. In addition, by placing on the ECC die LDPC codecs of different characteristics and sizes more options become available for the system adaptation to the environmental aggression level. In this way the 3D memory structure can be split into multiple ECC memory blocks on which different ECC mechanisms are applied when sensing that part of memory requires a more/less powerful protection.

6.3 LDPC Decoder Design Space

In the following we give a brief overview of LDPC decoding implementations and analyze them from different angles: *factor graph structure*, *algorithm*, *computation paradigm*, *architecture*, and *scheduling strategy*. Based on this evaluation we asses their applicability degree in the context of the proposed memory error correction framework.

Based on the graph structure, there are two LDPC code variants: (i) *regular* (Check Nodes (CNs) and Bit Nodes (BNs) degrees are constant) and (ii) *irregular* (otherwise). The *irregular* codes are asymptotically better than *regular* codes [21]. However, at finite-lengths irregular codes exhibit a higher error-floor phenomenon that may prevent their use in applications requiring very low error rates (e.g. memories). In addition, regular factor implementations require less interconnection complexity, resulting in reduced chip real estate and consuming less power.

LDPC decoding algorithms operate based on either *hard-decision* or *soft-decision*. Hard-decision algorithms process messages based on the fact that each information bit is either 1 or 0 (hard-information) [20], while soft-decision algorithms not only know the value of each information bit but also

the probability that reflects the reality, i.e., the bit value before being perturbed within the transmission channel. In view of the extra available knowledge, the decoding performance of soft-decision based decoders is higher at the expense of higher hardware requirements.

Among existing *soft-decision* algorithms, the Belief-Propagation (BP) (known also as Sum-Product (SP)) algorithm is the best in terms of decoding performance [22]. However, the high computational complexity of hyperbolic tangent functions, numerical instability, and the requirement of knowing the channel characteristics preclude effective BP based decoder implementations [23]. In an attempt to alleviate this issue the Min-Sum (MS) algorithm, which makes use of max-log approximations at the cost of decoding performance degradation, was proposed [24]. The MS decoding overhead can be also reduced by detecting and nullifying unreliable messages, which is the key feature of the Self-Corrected Min-Sum (SCMS) algorithm [25].

For memory error correction *soft-decision* algorithms are the first choice since they exhibit (i) higher decoding performance, which is the key factor towards high error resilience, and (ii) faster convergence, which is a crucial contributor to scrubbing and/or memory access time. However these advantages come at the expense of a larger real estate utilization when compared to the *hard-decision* counterpart and can only be obtained if soft-information can be extracted from the ECC protected memory circuits. To this end SR accesses should be implemented as soft-reads, i.e., instead of reading a memory location as a logic 0 or 1 its status is quantized into multiple bits by means of an Analog to Digital Converter (ADC) circuit. For the 3D memory placing the ADCs on the ECC die at the TSVs output instead of placing them inside the memory arrays is preferable to avoid memory footprint increase. Moreover, in this manner the ADCs are shared for the entire memory.

The hardware implementation of the LDPC decoding algorithms can rely on *conventional* or *unconventional* computing paradigm. In conventional computing, decoding algorithms are implemented either by means of floating-point or fixed-point arithmetic. Floating-point implementations perform better than fixed-point counterparts in terms of decoding performance at the expense of a higher hardware complexity. Alternatively, LDPC decoding may rely on the *unconventional* Stochastic Computing (SC) paradigm proposed by Gaines [26], Ribeiro [27], and Poppelbaum et al. [28] in 1967. For memory protection, *conventional* fixed point implementations are more appropriate since they provide a good balance between decoding performance and hardware complexity, which is important for securing memory error resilience at the expense of a

		Area	Latency	EC capability	Remarks
Factor Graph	Regular	+	+ *	-	* Clock speed
	Irregular	-	-	+	
Algorithm	Soft-decision	-	+ **	+	** Number of iterations
	Hard-decision	+	-	-	
Computational Implementation	Conventional	-	+ **	+	** Number of iterations
	Unconventional	+	-	-	Stochastic Computing
Architecture	Fully Parallel	-	+	=	Same EC capability
	Partly Parallel	+	-		
Scheduling	Flooding	-	+ ***	-	*** Latency per iteration
	Layered	+	-	+	

Note: + is better, - is worse

Figure 6.4: LDPC decoders summary.

reasonable hardware cost. We note however that due to its intrinsic fault tolerance SC based decoders could be of interest especially for protecting memories operating within harsh environmental conditions.

From the architecture perspective LDPC decoding implementation can be classified in *fully parallel* and *partially parallel*. *Fully parallel* ones achieve a high throughput decoding at the expense of a high area consumption. To reduce the hardware overhead, *partially parallel* implementations allow for hardware sharing/reuse. In this way, only a part of the graph is executed at a time on limited computation resources and memory is needed for temporary storage of messages passing between different parts of the factor graph. As a result, such an approach makes use of less chip real estate but provides a lower throughput than the fully parallel version. For memory protection *partially parallel* architectures seems the most appropriate since they enable a good balance between throughput and silicon area, which is important for providing a cost-effective memory with acceptable speed and error resilience.

Related to the scheduling strategy, there are mainly two flavors: *flooding* and *layered*. In *flooding* scheduling the Variable Nodes (VNs) and CNs are processed in a parallel fashion [29], while in *layered* scheduling groups of nodes are sequentially processed, which enables the immediate propagation of messages to neighboring nodes [30]. Generally, for cycle-free codes with infinitely many decoding iterations both types provide the same decoding performance. However, if the decoding iterations number is relatively small (e.g., practical

value), then the layered scheduling approach provides better performance than the flooding one. However, although the layered scheduling has less number of iterations, it requires a longer latency per iteration due to serialization in updating messages. For LDPC-based memory protection the *flooding* scheduling is a good choice for high speed memories due to its higher throughput, while for cost-effective memory, the layered scheduling is a preferable one due to its light-weight implementation and its superiority of EC capability.

In Figure 6.4 we present a comparison summary of the above mentioned LDPC approaches in terms of hardware area, decoding latency, and error correction capability (i.e., EC performance). Note that the error correction capability does not depend on the architectural choice. One can observe in Figure 6.4 that the conventional fixed-point implementation of soft-decision decoding with partially parallel architecture under flooded scheduling for regular LDPC codes is a preferable choice for cost-effective memory. However, in the context of high speed memory, fully parallel architecture is a good decision.

In the next section, the decoding performance of ECC-enhanced memory with some selected LDPC decoder implementations in comparison to the conventional memory is presented.

6.4 Evaluation

To evaluate the upper bound performance of our proposal, we simulate memories protected by: (i) state of the art (64 data bits and 8 check bits) extended Hamming (**SECDED**), and (ii) the proposed LDPC-based mechanism with (512, 64), (1024, 128), (2048, 256), and (4096, 512) codeword sizes. For LDPC we consider Quasi-Cyclic (QC) codes with variable-nodes degree $d_v = 4$ and check-nodes degree $d_c = 36$, which make use of Layered Min-Sum (**LMS**), Flooded Min-Sum (**FMS**), Layered Self-Corrected Min-Sum (**LSCMS**), and Flooded Self-Corrected Min-Sum (**FSCMS**) decoders operating on 3-bit and 4-bit soft information inputs and internal exchanged messages. To simulate memory fault occurrence, we use a binary symmetric channel model with various crossover probabilities (α). The decoders performances in terms of WER are plotted in Figure 6.5. It can be noticed that: (i) when a $WER = 10^{-6}$ is targeted, the SECDED protected memory tolerates memory faults up to $\alpha = 1 \times 10^{-5}$ while the LDPC protected ones bear up to 8×10^{-2}, (ii) utilizing soft information as input significantly improves the decoding performance, and, (iii) the longer the employed LDPC code the higher the obtained benefit.

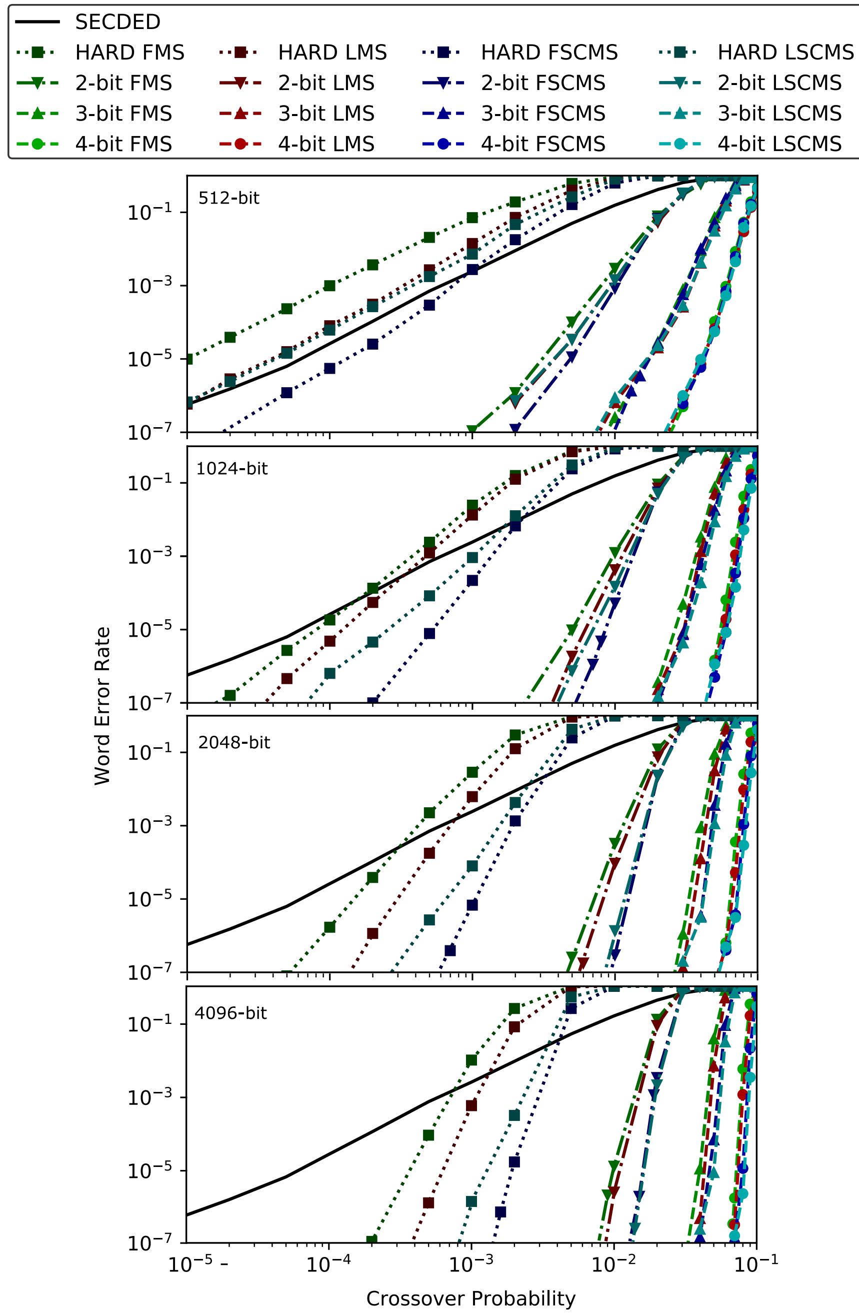


Figure 6.5: LDPC vs extended Hamming.

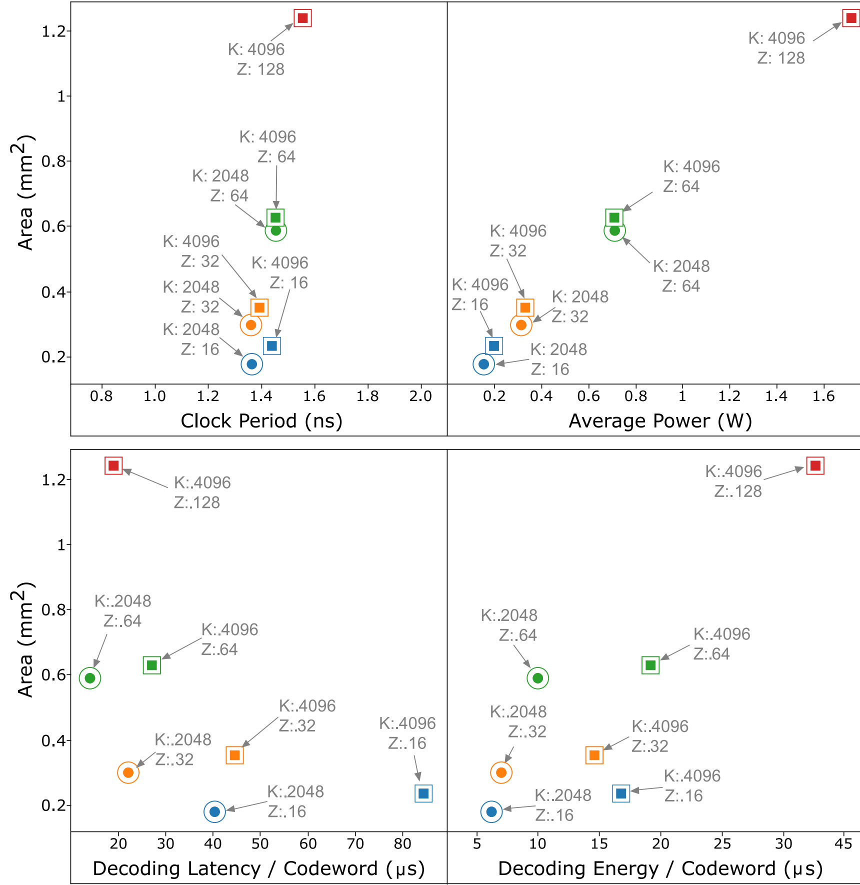


Figure 6.6: Area vs. clock period, area vs. power, area vs. decoding latency, and area vs. decoding energy.

Table 6.1: LDPC Encoders and Decoders in ASIC Implementations

	Energy/Bit (pJ/Bit)		Latency/Bit (ps/Bit)		Area/Bit (um^2/Bit)	
Instances	Dec	Enc	Dec	Enc	Dec	Enc
K2048_Z16	3041.99	0.19	19702.15	2.19	87.89	57.96
K2048_Z32	3417.96	0.19	10849.61	2.19	146.48	57.72
K2048_Z64	4877.93	0.20	6875.00	2.19	288.08	60.32
K4096_Z16	4077.14	0.41	20546.88	1.09	58.59	176.34
K4096_Z32	3549.80	0.41	10834.96	1.09	85.44	175.75
K4096_Z64	4667.96	0.40	6611.32	1.09	153.80	172.03
K4096_Z128	7929.68	0.41	4626.46	1.09	302.73	177.27

Memory Size (MB)	Memory Footprint (mm^2)	LDPC Decoder & Encoder Utilization (%)		Free Area Available on the EC die (mm^2)		Additional Decoders that can be Accomodated (#)	
		2048	4096	2048	4096	2048	4096
1	2.44	24	40	1.86	1.48	10	6
2	3.13	19	30	2.55	2.17	14	9
4	4.43	13	21	3.85	3.47	21	14
8	7.06	8	13	6.48	6.09	36	25

Figure 6.7: LDPC error correction die real estate utilization.

Physical synthesis on a commercial 40nm low-power CMOS was performed using Cadence Encounter RTL Compiler [31] for (2048, 256) and (4096, 512) codecs with their designs being automatically generated as in [32]. Various QC LDPC codes with different sub-matrix sizes (Z) were considered. The synthesis results are presented in Figure 6.6 and Table 6.1 from which one can observe that: (i) the sub-matrix size Z is the main design parameter from the hardware cost point of view, (ii) the decoder **K4096_Z16** is one of the most balanced options, (iii) the energy and the latency needed for encoding are 3 to 4 orders of magnitude smaller when compared to decoding, (iv) the encoding energy, latency, and area figures are relatively Z independent, and, (v) encoding smaller codes is more energy and area efficient at the cost of a longer latency per bit. In Figure 6.7 we provide an LDPC (**K2048/4096 Z16**) tier real estate utilization estimation for 4-dies 3D memories. Memory footprints are obtained with a modified Cacti 6 simulator [33] for 40nm CMOS technology as in [16]. We mention that the 2048-bit LDPC protection is assured by means of two parallel scrubbing blocks, which each having allocated half of the memory size.

Next, we performed experiments on memory access traces considering a system based on an ARMv7-A processor with a two level cache hierarchy: (i) 32-kB instruction and data L1 caches, and (ii) a unified 4-MB L2 cache implemented as a 4 die 3D memory. On each L2 cache die there are 8 sub-banks (with an 512-bit I/O width), each being composed of 8 mats (with an 64-bit I/O width). We simulated the system with a modified gem5 simulator [34] in order to obtain access traces for the L2 cache when running a memory intensive SPEC CPU2000 [35] suite benchmarks subset. We run each benchmark for 1 million committed instructions and we injected transient faults into the memory at random data locations during each cycle. In Figure 6.8 we present: (i) the total number of erroneous 64-bit words which become visible to the executed application on average per run, and, (ii) the total number of runs when no error was propagated. It can be noticed from Figure 6.8 that the **SECDED** protection manages to significantly reduce the number of propagated 64-bit erro-

		64b Erroneous Propagated Words (Average)				Clean Runs			
				LDPC				LDPC	
Benchmark	Alpha	NO EC	SECDED	2048	4096	NO EC	SECDED	2048	4096
gzip	6×10^{-2}	7816.01	7439.35	4186.59	6059.53	0	0	0	0
	5×10^{-2}	8906.20	8129.51	786.24	326.88	0	0	0	0
	4×10^{-2}	9728.93	8244.60	8.55	0.09	0	0	627	1977
	3×10^{-2}	9813.73	7285.47	0.01	0	0	0	1996	2000
	10^{-2}	4051.61	1374.14	0	0	0	0	2000	2000
	10^{-3}	77.81	2.90	0	0	0	1177	2000	2000
	10^{-4}	0.81	0.01	0	0	1732	1998	2000	2000
gcc	6×10^{-2}	465.90	443.45	248.68	361.05	0	0	0	0
	5×10^{-2}	422.86	386.18	37.97	15.88	0	0	5	243
	4×10^{-2}	362.39	307.34	0.33	0	0	0	1913	2000
	3×10^{-2}	280.47	208.05	0	0	0	0	2000	2000
	10^{-2}	64.65	21.91	0	0	0	0	2000	2000
	10^{-3}	0.95	0.04	0	0	979	1919	2000	2000
	10^{-4}	0.01	0	0	0	1985	2000	2000	2000
mcf	6×10^{-2}	446.36	424.95	238.13	345.24	0	0	0	0
	5×10^{-2}	365.20	333.23	32.28	13.48	0	0	35	344
	4×10^{-2}	285.22	241.93	0.23	0	0	0	1940	2000
	3×10^{-2}	199.80	148.44	0	0	0	0	2000	2000
	10^{-2}	37.63	12.80	0	0	0	0	2000	2000
	10^{-3}	0.50	0.02	0	0	1356	1969	2000	2000
	10^{-4}	0.01	0	0	0	1986	2000	2000	2000

Figure 6.8: L2 correction capability.

neous words only for $\alpha < 10^{-3}$ while **LDPC** protection significantly reduces the number of 64-bit erroneous propagated words already at $\alpha = 5 \times 10^{-2}$.

6.5 Conclusion

In this chapter we proposed and investigated an adaptive LDPC-based memory error correction mechanism best suited for 3D memories and potentially applicable to a large memory technologies range. Our results indicate that our proposed LDPC based memory protection offers substantially improved error correction capabilities when compared to state of the art extended Hamming ECC, being able to assure correct application execution for $\alpha < 3 \times 10^{-2}$ memory error rates.

Note. The content of this chapter is based on the following paper:

M. Lefter, G.R. Voicu, T. Marconi, V. Savin, S.D. Cotofana, **LDPC-Based Adaptive Multi-Error Correction for 3D Memories**, *IEEE International Conference on Computer Design (ICCD)*, Boston, USA, November 2017.

Bibliography

[1] S. Nassif, “The light at the end of the cmos tunnel,” in *ASAP*, 2010.

[2] S. Rusu, M. Sachdev, C. Svensson, and B. Nauta, “T3: Trends and challenges in vlsi technology scaling towards 100nm,” in *ASP-DAC*, 2002.

[3] G. Georgakos, P. Huber, M. Ostermayr, E. Amirante, and F. Ruckerbauer, “Investigation of increased multi-bit failure rate due to neutron induced seu in advanced embedded srams,” in *VLSI*, 2007.

[4] A. Dixit and A. Wood, “The impact of new technology on soft error rates,” in *IRPS*, 2011.

[5] E. Ibe, H. Taniguchi, Y. Yahagi, K. Shimbo, and T. Toba, “Impact of scaling on neutron-induced soft error in srams from a 250 nm to a 22 nm design rule,” *ED*, 2010.

[6] T. Suzuki, Y. Yamagami, I. Hatanaka, A. Shibayama, H. Akamatsu, and H. Yamauchi, “0.3 to 1.5v embedded sram with device-fluctuation-tolerant access-control and cosmic-ray-immune hidden-ecc scheme,” in *ISSCC*, 2005.

[7] R. Naseer and J. Draper, “Parallel double error correcting code design to mitigate multi-bit upsets in srams,” in *ESSCIRC*, 2008.

[8] M. Horiguchi and K. Itoh, *Nanoscale Memory Repair*. 2011.

[9] P. Garrou, C. Bower, and P. Ramm, *3D Integration: Technology and Applications*. 2008.

[10] B. Black, M. Annavaram, N. Brekelbaum, J. DeVale, L. Jiang, G. H. Loh, D. McCaule, P. Morrow, D. W. Nelson, D. Pantuso, P. Reed, J. Rupley, S. Shankar, J. Shen, and C. Webb, “Die stacking (3d) microarchitecture,” in *MICRO*, 2006.

[11] J. Zhao, Q. Zou, and Y. Xie, “Overview of 3d architecture design opportunities and techniques,” *DT*, 2017.

[12] B. Prince, *Vertical 3D Memory Technologies* . 2014.

[13] S. Safiruddin, D. Borodin, M. Lefter, G. Voicu, and S. D. Cotofana, “Is 3d integration the way to future dependable computing platforms?,” in *OPTIM*, 2012.

[14] M. Lefter, G. Voicu, M. Taouil, M. Enachescu, S. Hamdioui, and S. D. Cotofana, “Is tsv-based 3d integration suitable for inter-die memory repair?,” in *DATE*, 2013.

[15] H. M. C. Consortium, “Hybrid memory cube specification 2.1,” 2014.

[16] M. Lefter, G. Voicu, and S. Dan Cotofana, “A shared polyhedral cache for 3d wide-i/o multi-core computing platforms,” in *ISCAS*, 2015.

[17] JEDEC, “High bandwidth memory,” 2016.

[18] D. U. Lee, K. W. Kim, K. W. Kim, H. Kim, J. Y. Kim, Y. J. Park, J. H. Kim, D. S. Kim, H. B. Park, J. W. Shin, J. H. Cho, K. H. Kwon, M. J. Kim, J. Lee, K. W. Park, B. Chung, and S. Hong, “25.2 a 1.2v 8gb 8-channel 128gb/s high-bandwidth memory (hbm) stacked dram with effective microbump i/o test methods using 29nm process and tsv,” in *ISSCC*, 2014.

[19] M. Lefter, T. Marconi, G. Voicu, and S. D. Cotofana, "Low cost multi-error correction for 3d polyhedral memories," in *NANOARCH*, 2017.

[20] R. Gallager, "Low-density parity-check codes," *TIT*, 1962.

[21] B. Vasic, S. K. Chilappagari, and D. V. Nguyen, "Chapter 6 - failures and error floors of iterative decoders," in *Academic Press Library in Mobile and Wireless Communications*, 2014.

[22] F. Kschischang, B. Frey, and H.Loeliger, "Factor graphs and the sum-product algorithm.," in *TIT*, 2001.

[23] V. Savin, "Chapter 4 - LDPC decoders," in *Academic Press Library in Mobile and Wireless Communications* (D. Declerq, M. Fossorier, and E. Biglieri, eds.), 2014.

[24] M. P. C. Fossorier, M. Mihaljevic, and H. Imai, "Reduced complexity iterative decoding of low-density parity check codes based on belief propagation," *TC*, 1999.

[25] V. Savin, "Self-corrected min-sum decoding of ldpc codes," in *ISIT*, 2008.

[26] B. R. Gaines, "Stochastic computing," in *SJCC*, 1967.

[27] S. T. Ribeiro, "Random-pulse machines," *IEEE-TC*, 1967.

[28] W. J. Poppelbaum, C. Afuso, and J. W. Esch, "Stochastic computing elements and systems," in *FJCC*, 1967.

[29] F. R. Kschischang and B. J. Frey, "Iterative decoding of compound codes by probability propagation in graphical models," *IEEE JSAC*, 1998.

[30] D. E. Hocevar, "A reduced complexity decoder architecture via layered decoding of ldpc codes," in *WSPS*, 2004.

[31] Cadence, "Cadence Encounter RTL Compiler product description," 2016.

[32] O. Boncalo, P. F. Mihancea, and A. Amaricai, "Template-based qc-ldpc decoder architecture generation," in *ICICSP*, 2015.

[33] N. Muralimanohar, R. Balasubramonian, and N. P. Jouppi, "CACTI 6.0: A tool to model large caches," tech. rep., HP Laboratories, 2009.

[34] N. Binkert, B. Beckmann, G. Black, S. K. Reinhardt, A. Saidi, A. Basu, J. Hestness, D. R. Hower, T. Krishna, S. Sardashti, R. Sen, K. Sewell, M. Shoaib, N. Vaish, M. D. Hill, and D. A. Wood, "The gem5 simulator," *SIGARCH*, 2011.

[35] J. L. Henning, "SPEC CPU2000: measuring CPU performance in the new millennium," *Computer*, 2000.

7
Conclusions and Future Work

Technology feature size scaling represented the key to improved computing systems performance for decades. However, once entering the deep nanometer range, due to physical and quantum mechanical effects, technology scaling reached the limit where increasing clock frequency was no longer sustainable for delivering the expected improvements. To cope with this issue computer designers advocated for a paradigm shift from uniprocessors to multiprocessor micro-architectures, which lead to an increased amount of simultaneous parallel memory accesses. Hence, ensuring data availability became more important than reducing functional units latency, and complex memory hierarchies emerged as a result. Additionally, due to device miniaturization power consumption and reliability become of high importance. In view of the above, in this thesis we investigated alternative avenues to foster emerging 3D Stacked Integrated Circuit (3D-SIC) technology potential in improving memory hierarchies performance. We pursued the following three main research lines: (i) increase data availability, (ii) diminish power consumption, and, (iii) improve reliability and dependability.

This chapter is organized in two sections as follows: in Section 7.1 we present an overview of the thesis by summarizing its major contributions, while in Section 7.2 we propose future research directions.

7.1 Summary

The work presented in this thesis can be summarized as follows.

In **Chapter 2** we introduced a multi-port memory design consisting of multiple banks stacked on top of each other, forming a polyhedral structure, which exhibits an enriched access mechanism set with a reduced bank conflict prob-

ability, similar to true multi-port designs. TSV bundles are distributed over the entire memory footprint and are selectively connected to the internal data lines in every sub-bank in order to support vertical wide-I/O interface memory accesses. In addition, the TSV bundles facilitate horizontal port access to/from a non-local bank (located on a different die), such that: (i) a large number of virtual banked ports are created, greater than the number of physical banks in the memory, and, (ii) wide internal data transfers can be easily performed between different memory banks. Furthermore, an identical layout structure of all memory dies is proposed, which reduces the manufacturing cost. We evaluated the potential of our proposal by considering its utilization as a shared last level cache in a TSV-based wide-I/O multi-core system. The interface with the upper level caches on the CPU-side is realized horizontally, while the DRAM interface is realized vertically through the TSV bundles that traverse the entire IC stack. The wide-I/O interface increases the communication bandwidth with the main memory while the large number of virtual ports ensure an enhanced data availability for the processing cores. We compared our polyhedral cache against traditional banked multi-port 2D implementations, considering relevant metrics: (i) access time, (ii) footprint, and (iii) leakage and dynamic energy consumption per access cycle. Our results indicate that the 3D polyhedral cache access time is in general smaller than the one of its planar counterparts, with up to 50% reduction in the best case. The inherent footprint reduction directly influences the active energy consumption, which in same cases is decreased almost $8\times$ for an 8-die 8 MB cache.

In **Chapter 3** we proposed a novel low power dual port memory, which relies on the synergistic utilization of: (i) NEMFET based short circuit current free inverters for data storage, and (ii) versatile CMOS based logic for read and write operations, and data preservation. TSV-based 3D stacking technology is employed for construction of the memory structure since it smoothly facilitates the co-integration of NEM and conventional CMOS devices, which, for the time being, appears not to be feasible on the same tier. Hence, in the proposed memory organization, the NEMFET-based storage elements reside on one tier, while the CMOS logic required to retrieve, maintain, and write the data is located on another tier. By utilising only one inverter per memory cell, instead of a cross coupled pair, a low write energy is achieved, as only one bitline is required. Furthermore, the static energy is drastically reduced due to NEMFET's extremely low off current. Next, we evaluated the utilization of energy effective 3D-Stacked Hybrid NEMFET-CMOS memories as replacements for traditional CMOS based SRAMs in L1 and L2 processor caches. This solution provides two orders of magnitude cache static energy reduction,

albeit a slightly increased dynamic energy consumption, and an approximately 55% larger footprint. NEMFET's larger size makes the inverter-based memory cell a more powerful driver, with a direct impact on the read access time, which is on average 7% smaller when compared to CMOS implementations. The write access time is with about 3 ns higher, as it is dominated by the mechanical movement of the NEMFET's suspended gate. To identify if the write latency overhead has any impact on processor performance, measured in terms of Instructions per-Cycle (IPC), we considered as evaluation vehicle a state of the art mobile out-of-order processor core. Our simulations on a set of SPEC 2000 benchmarks indicate that the extra write latency impact on the overall system performance is rather low or even negligible for L1 instruction and L2 cache instances, while for L1 data caches IPC decreases between 1% to 12%. On the same benchmark set we evaluated the energy benefits of utilizing 3D-Stacked Hybrid NEMFET-CMOS caches. Our experiments indicate that, in spite of their increased write access time and dynamic energy, 3D Stacked Hybrid NEMFET-CMOS L2 caches provide substantial energy savings, i.e., 38 % total caches energy reduction on the average.

In **Chapter 4** we addressed low level issues related to memory repair and we investigated for current and foreseeable technology developments the actual potential of using repair in the vertical dimension by accessing unutilized spare resources available on neighboring stacked dies. For this we proposed a memory repair framework based on a provider-consumer pair scenario consisting of several possible implementation schemes that allow consumers when in need to make use of the available row/columns redundant resources on the providers side. The proposed infrastructure handles the data remapping from provider to consumer in a transparent manner without interfering with the providers operation. Our analysis suggests that at current state-of-the-art TSV dimensions inter-die repair schemes using column sharing could have a positive contribution to the overall memory reparability mechanism at the expense of a reasonable area overhead. For row repair, however, most memory configurations require TSV dimensions to scale down at least with one order of magnitude in order to make this approach a possible candidate for its inclusion in 3D memory systems. We also performed a theoretical analysis of the proposed 3D repair schemes implications on memory access time, which indicates that no substantial delay overhead is expected.

In **Chapter 5** we introduced two novel error correction mechanisms for protecting memories from Multiple Bit Upsets (MBUs). Our proposals rely on the utilization of Bose-Chaudhuri-Hocquenghem (BCH) codes which, by operating on codewords that are larger than the ones employed by state of the

art extended Hamming ECC for single error correction, allow for Multi-Error Correction (MEC) capabilities. For the first mechanism we propose to physically place the codecs coplanar with the memory dies. In this manner data words and associated check bits can be concatenated in up to h-bit codewords, where h is the sub-bank I/O width, and each die includes several MEC encoders/decoders. The second mechanism further exploits the polyhedral memories organization, whose inherently fast and customizable wide-I/O vertical access mechanisms allow for the codeword creation with data and check bits from multiple memory sub-banks. Those long codewords can be smoothly transferred to/from a dedicated error correction die, where a variable codecs number could be placed. The proposed approaches can operate independently or in synergy, leading to the creation of a large design space from where the most appropriate scheme can be chosen by balancing the desired error correction capability while fulfilling different overhead budgets in terms of memory footprint, latency, and energy. As main operation mode we propose an online scrubbing procedure that performs the error detection and correction decoupled from the normal memory operation. However, on the fly write and read operations can be also performed if environmental conditions require as such. We evaluated our proposals by considering a 4-die 4-MB polyhedral memory as a case study and evaluated by means of simulation the efficiency of various data width codes implementations. The results indicate that our proposal outperforms state of the art extended Hamming SEC in terms of error correction capability, being able to diminish the Word Error Rates (WER) by 2 to 12 orders of magnitude for bit error probabilities between 10^{-4} and 10^{-6}, while requiring less redundancy overhead. When on the fly error correction and detection is performed, our MEC approaches generate up to 40% and 80% write and read latency increases, respectively, when compared to Hamming ECC. While no extra energy consumption was noticed for read operations irrespective of the codeword size, this was not the case for write operations for which up to 9× increases were measured for the longest codec instances. However, by relocating the codecs on a specialized die in the 3D memory stack and by enabling the parallel energy effective scrubbing procedure, we managed to hide the codec latency and provided 10% and 25% write and read latency reductions, respectively, and 13% footprint reduction.

In **Chapter 6** we followed a similar line of reasoning as in Chapter 5, but considered the utilization of Low Density Parity Check (LDPC) codes, whose performances in terms of error correction are close to the Shannon limit, to further boost the MEC potential. In the proposed mechanism, extended codewords that are larger than the memory access granularity are formed by com-

bining data and check bits from multiple memory sources. They are next transfered on/from a dedicated ECC die by making use of the available numerous TSV bundles, where a number of iterative LDCP codecs perform encoding and decoding. In order to hide the additional non constant error correction time required by LDPC codecs, an adaptive online scrubbing policy performs memory maintenance without interfering with data requests issued by the System on Chip (SoC) computation cores. The scrubbing policy takes into account possible check bits invalidations during write operations that may occur due to the utilization of codewords larger than the memory addressing width. In addition, the modus operandi of the memory correction mechanism can be adapted to the environmental conditions if necessary, by performing on the fly LDPC encoding/decoding. However, this requires interference with memory operations and has a detrimental impact on performance in terms of throughput, latency and energy consumption. We performed a design space exploration in which we identified conventional fixed-point soft-decision decoding with partially or fully parallel architecture under flooded scheduling LDPC decoders as most suitable for memory protection. To evaluate our proposal we considered 3D memories protected by the proposed mechanism with various data width codes implementations. Our simulations indicate that our proposal clearly outperforms state of the art ECC schemes with fault tolerance improvements by a $4710\times$ factor being obtained when compared to extended Hamming ECC. Furthermore, we evaluated instances of the proposed memory concept equipped with different LDPC codecs implemented on a commercial 40nm low-power CMOS technology and evaluate them on actual memory traces in terms of error correction capability, area, latency, and energy. Our results indicate that our proposed LDPC based memory protection offers substantially improved error correction capabilities when compared to state of the art extended Hamming ECC, being able to assure correct application execution for $\alpha < 3 \times 10^{-2}$ memory error rates.

7.2 Future Research Directions

Subsequently, we outline the following directions as a suggested continuation of the research undertaken in this thesis.

1. With respect to the proposed polyhedral memory design, further investigations by making use of a processor simulator instrumented in order to properly capture memory accesses, would provide additional insights into the actual benefits that are to be expected at the system level when

employing polyhedral memories in a more closer to real life scenario. An analysis that would determine the optimum number of ports and data I/O width could be further performed in this manner. In addition, following a similar simulation flow, a comparison against other state-of-the-art cache implementations that follow a network on a chip approach with non uniform access could be completed.

2. In this thesis we proved the low power efficiency and utilization potential of the proposed hybrid NEMFET-CMOS dual port memory. However, we did not perform a thorough evaluation of the TSV size implications on the memory cell, an action point that is of great interest, since TSV shrinking follows a completely different agenda than the CMOS/NEMFET device one. Regarding the NEMFET large size, it is expected that it would also go through a shrinking process. Hence, it would be interesting to estimate the NEMFET scaling process implications on the proposed memory. In addition, in the evaluations we have compared the hybrid memory proposal against CMOS only implementations. A more comprehensive comparison would include also other emerging memory alternatives, e.g., MRAM, PCRAM, etc.

3. In terms of memory repair, our current evaluation considers sharing redundant resources only between two adjacent dies, since the overhead of allowing for spare replacement between multiple dies seems very high. It would be interesting to investigate the actual complexity of such a multiple dies redundancy sharing approach from implementation as well as from yield improvement perspective.

4. Regarding the multi-error correction mechanisms proposed in this thesis, a more detailed analysis that would take into consideration multiple codecs of different sizes and types available for the adaptive mechanism could be of interest.

List of Publications

Publications related to the dissertation

International Conferences

1. M. Lefter, G.R. Voicu, M. Taouil, M. Enachescu, S. Hamdioui, S.D. Cotofana, **Is TSV-based 3D Integration Suitable for Inter-die Memory Repair?**, *Design, Automation & Test in Europe (DATE)*, Grenoble, France, March 2013.
2. M. Lefter, M. Enachescu, G.R. Voicu, S.D. Cotofana, **Energy Effective 3D Stacked Hybrid NEMFET-CMOS Caches**, *ACM/IEEE International Symposium on Nanoscale Architectures (NANOARCH)*, Paris, France, March 2014.
3. M. Lefter, G.R. Voicu, S.D. Cotofana, **A Shared Polyhedral Cache for 3D Wide-I/O Multi-Core Computing Platforms**, *IEEE International Symposium on Circuits and Systems (ISCAS)*, Lisbon, Portugal, May 2015.
4. M. Lefter, T. Marconi, G.R. Voicu, S.D. Cotofana, **Low Cost Multi-Error Correction for 3D Polyhedral Memories**, *IEEE/ACM International Symposium on Nanoscale Architectures (NANOARCH)*, Newport, USA, July 2017.
5. M. Lefter, G.R. Voicu, T. Marconi, V. Savin, S.D. Cotofana, **LDPC-Based Adaptive Multi-Error Correction for 3D Memories**, *IEEE International Conference on Computer Design (ICCD)*, Boston, USA, November 2017.

International Journals

1. M. Enachescu, M. Lefter, G.R. Voicu, S.D. Cotofana, **Low-Leakage 3D Stacked Hybrid NEMFET-CMOS Dual Port Memory**, *IEEE Transactions on Emerging Topics in Computing*, July 2016.

Other publications

International Conferences

1. S. Safiruddin, D. Borodin, M. Lefter, G.R. Voicu, S.D. Cotofana, **Is 3D Integration The Way to Future Dependable Computing Platforms?**, *International Conference on Optimization of Electrical and Electronic Equipment (OPTIM)*, Brasov, Romania, May 2012.

2. S. Safiruddin, M. Lefter, D. Borodin, G.R. Voicu, S.D. Cotofana, **Zero-Performance-Overhead Online Fault Detection and Diagnosis in 3D Stacked Integrated Circuits**, *IEEE/ACM International Symposium on Nanoscale Architectures (NANOARCH)*, Amsterdam, The Netherlands, July 2012.

3. M. Taouil, M. Lefter, S. Hamdioui, **Exploring Test Opportunities for Memory and Interconnects in 3D ICs**, *International Design & Test Symposium (IDT)*, Doha, Qatar, December 2012.

4. M. Enachescu, M. Lefter, A. Bazigos, A. Ionescu, S.D. Cotofana, **Ultra Low Power NEMFET Based Logic**, *IEEE International Symposium on Circuits and Systems (ISCAS)*, Beijing, China, May 2013.

5. G.R. Voicu, M. Lefter, M. Enachescu, S.D. Cotofana, **3D Stacked Wide-Operand Adders: A Case Study**, *IEEE International Conference on Application-Specific Systems, Architectures and Processors (ASAP)*, Washington D.C., USA, June 2013.

Samenvatting

In de afgelopen halve eeuw maakten Integrated Circuits (IC's) een agressieve, prestatiegerichte technologische schaling mee. Naarmate de technologie verder schaalde in het nanometergebied, werden fysische en kwantummechanische effecten die voorheen irrelevant waren invloedrijk of zelfs dominant, wat bijvoorbeeld resulteerde in niet langer verwaarloosbare lekstromen. Bij pogingen om dergelijke kleine geometrische dimensies volgens een patroon te maken, nam de variabiliteit van technologische parameters aanzienlijk toe. Bovendien wordt het moeilijker om op betrouwbare wijze een dergelijk groot aantal kleine transistoren te verwerken en te integreren in grootschalige IC's, aangezien er rekening moest worden gehouden met een substantiële toename in vermogensdichtheid. Schaling geïnduceerde prestatie was niet langer voldoende meer om aan de verwachte verbeteringen te voldoen, wat leidde tot een paradigma-omschakeling van uniprocessors naar micro-architecturen met meerdere processors. Tegelijkertijd, omdat voor bepaalde toepassingsdomeinen zoals big data en internet of things de verwerkte hoeveelheid gegevens substantieel toeneemt, zijn computerontwerpers meer bezig met het waarborgen van databeschikbaarheid dan met het verminderen van de latentie van functionele eenheden. Als gevolg hiervan maken de meest geavanceerde computersystemen gebruik van complexe geheugenhiërarchieën, bestaande uit maximaal vier cacheniveaus met meerdere gedeelde schema's, waardoor het geheugen een dominant ontwerpelement is dat de algehele systeemprestaties en correct gedrag aanzienlijk beïnvloedt. In deze context komt 3D Stacked Integrated Circuit (3D SIC) technologie naar voren als een veelbelovende methode om nieuwe ontwerpmogelijkheden mogelijk te maken, aangezien deze de middelen biedt om chips met elkaar te verbinden met korte verticale draden. In dit proefschrift behandelen we de bovengenoemde uitdagingen met betrekking tot het geheugen door het gebruik van 3D SIC technologie in geheugenontwerpen te onderzoeken. Ten eerste stellen we een nieuw multi-port polyhedraal geheugen voor dat een verrijkt toegangsmechanisme biedt met een zeer laag conflict rate en we evalueren het potentieel in gedeelde caches. Ten tweede stellen we een laag vermogen hybride geheugen voor waarin 3D technologie zorgt voor een soepele cointegratie van: (i) op kortsluitstroom vrije Nano-elektromechanische veldeffecttransistor (NEMFET) gebaseerde inverters voor gegevensopslag, en (ii) op

CMOS-gebaseerde logica voor lees- / schrijfbewerkingen en gegevensbehoud. Ten derde stellen we een geheugen herstel framework voor dat gebruik maakt van de 3D verticale nabijheid voor het uitwisselen van redundante resources tussen chips. Ten slotte stellen we nieuwe schema's voor, voor het uitvoeren van gebruikerstransparante multi-fout correctie en detectie, met dezelfde of zelfs een lagere redundantie dan die vereist is voor geavanceerde Hamming correctieschema's met enkelvoudige fouten.

Propositions

accompanying the PhD dissertation

On Leveraging Vertical Proximity in 3D Memory Hierarchies

---

1. To flatten the memory hierarchy we have to go vertically. [This Thesis]
2. The more the better is not true when it concerns the number of layers in a three dimensional circuit. [This Thesis]
3. More than Moore revolves around a large variety of tradeoffs that emerge from heterogeneous integration. [This Thesis]
4. As seen from the societal behavior binary logic will never die.
5. Societies that favor group interest over individual rights rarely do not end up in a dictatorship.
6. The virus of equality urges us to fiercely seek our own evil.
   [Nicolae Steinhardt - Romanian writer (1912 - 1989)]
7. The freedom to fly involves the freedom to fall, i.e., there are no rights without responsibilities.
8. For scarcely for a righteous man will one die: yet peradventure for a good man some would even dare to die.
   [Romans 5,7]
9. Theories are patterns without value. What counts is action.
   [Constantin Brancusi - Romanian sculptor (1876 - 1957)]
10. To see the end of your PhD is one thing, finishing it is another.

These propositions are regarded as opposable and defendable, and have been approved as such by the promotor, Prof. dr. NameInitials Surname.

Curriculum Vitae

Mihai Lefter was born in Brașov, Romania, on the 10th of December, 1984. After graduating ”Grigore Moisil” Computer Science College, he started studying Automatics at Transilvania University. He received his Engineer Degree (equivalent with Master of Science under the Bologna process) in 2008. Later, he received his MSc degree in Computer Engineering from Delft University of Technology, in 2010. His master thesis, performed in collaboration with Cosine Research, focused on developing highly customizable FPGA trigger logic units for system-on-chip architectures employed in space electronics instrumentation. In the same year, he started working towards a PhD degree under the supervision of Dr. Sorin Coțofană, at the Computer Engineering laboratory of Delft University of Technology, The Netherlands. The topic of his research was novel memory hierarchies that leverage the opportunities provided by the emerging three dimensional through-silicon via integration technology. Starting 2016, Mihai works as a software engineer and lecturer in the Human Genetics department at Leiden University Medical Center.