NTNU – Trondheim
Norwegian University of
Science and Technology

Estimating the Energy Consumption of Emerging Random Access Memory Technologies

Magnus Moreau

Master of Science in Electronics
Submission date: July 2013
Supervisor: Thomas Tybell, IET
Co-supervisor: Alf Petter Syvertsen, Energy Micro AS

Norwegian University of Science and Technology
Department of Electronics and Telecommunications

Problem formulation

Every year, several billion microcontrollers are produced and used in a variety of products and settings. Increasingly many of these systems are battery operated, and are often placed in environments where replacing batteries is not an option. Increasing the battery lifetime for such systems is a high priority both for economic and environmental reasons.

Beside the power consumed by the processor, the power consumed by the memory system in the microcontroller is the major drain on battery life. In recent years a wide array of new exotic memory technologies has been developed including FeRAM, MRAM, STT-MRAM and PCRAM.

The goals of the master thesis are to:

1. Develop a model to estimate the power consumption for memory technologies likely to be commercially available by 2017 including, DRAM, SRAM, FeRAM, MRAM, STT-MRAM and PCRAM, in order to support a choice of future RAM technology in microcontrollers.
2. Sketch how to develop a measuring scheme in order to verify the developed theoretical model under point 1.

Thomas Tybell
Trondheim, June 2013

Abstract

In this work, a model for estimating the energy consumption of different types of random access memory(RAM) technologies, likely to be commercially available by 2017, has been developed. The goal for this model has been to evaluate which of the memory technologies that will be the most energy efficient in 2017. This was done by building the model on the required energies to read or write a bit for the different technologies. The memory technologies that have been modelled are: Dynamic RAM (DRAM), Static RAM (SRAM), Ferroelectric RAM (FeRAM), Magnetic RAM (MRAM), Spin-Torque Transfer Magnetic RAM (STT-MRAM) and Phase Change RAM (PCRAM).

The volatile memory technologies, DRAM and SRAM, have been estimated to have the lowest energy consumption if the memories are operated at a high duty cycle. However, if the duty cycle is reduced, the emerging non-volatile memory technologies become more energy efficient. The FeRAM was estimated to have the lowest power consumption when manufactured with the technology available today. And it has been estimated that with a duty cycle lower than 8.5×10^{-4}, FeRAM technology consumes less power than the SRAM, and with a duty cycle lower than 1.6×10^{-4}, the FeRAM consumes less power than DRAM. When looking forward towards 2017, STT-MRAM was estimated to have the lowest power consumption of the emerging memory technologies. It was estimated that the STT-MRAM consumes less power than SRAM for duty cycles lower than 2.3×10^{-2} and consumes less power than the DRAM for duty cycles lower than 4.6×10^{-4}.

An experimental set-up has been developed to validate the model, and a case study of some selected memories has been performed. With this case study some limitations on the theoretical model have been pointed out. These limitations are believed to be reduced if the memories are embedded on chip.

Sammendrag

I denne oppgaven er en modell, for å estimere energiforbruket til forskjellige typer «Random Access Memories»(RAM)-teknologier, som er forventet å være kommersielt tilgjengelig innen 2017, utviklet. Målet med modellen har vært å evaluere hvilken type minneteknologi som er den mest energivennlige i 2017. Dette ble gjort ved å basere modellen på de forskjellige energiene som må tilføres for å lese eller skrive et bit. Minneteknologiene som har blitt modellert er: Dynamisk RAM (DRAM) Statisk RAM (SRAM), Ferroelektrisk RAM, Magnetisk RAM (MRAM), Spin-Moment overførings Magnetisk RAM (STT-MRAM) og fase endrings RAM (PCRAM).

De volatile minneteknologiene DRAM og SRAM er estimert til å ha det laveste energiforbruket hvis minnene drives med en høy driftssyklus. På den andre siden, hvis driftssyklusen er redusert vil de ikke volatile minneteknologiene bli mer energieffektive. FeRAM var den minneteknologien som ble estimert til å ha det laveste energiforbruket hvis minnene produseres med teknologi som er tilgjengelig i dag. Det er estimert at ved en driftssyklus lavere enn 8.5×10^{-4}, så vil FeRAM bruke mindre energi enn SRAM, og ved en driftssyklus lavere enn 1.6×10^{-4} er FeRAM estimert til å ha et lavere energiforbruk enn DRAM. Når vi ser fram mot 2017 er det STT-MRAM som er estimert til å ha det laveste energiforbruket. Det ble anslått at STT-MRAM bruker mindre energi enn SRAM hvis driftssyklusen er lavere enn 2.3×10^{-2}, og bruker mindre energi enn DRAM for driftssykluser lavere enn 4.6×10^{-4}

Et måleoppsett ble utviklet for å validere modellene og en casestudie av noen utvalgte minner ble utført. Denne studien har påpekt noen begrensninger ved den teoretiske modellen. Disse begrensingene er antatt å bli redusert dersom minnene blir innebygget på brikken.

Preface

> "Essentially, all models are wrong, but some are useful"
>
> George E. P. Box

This is a master thesis, which has been carried out at the Department of Electronics and Telecommunications at NTNU, during the spring 2013. This thesis has been written as the final part of the master program at NTNU, and is a continuation of my project thesis [1]. The work was conducted in collaboration with Energy Micro AS.

Creating a model to estimate the energy consumption of random access memories(RAM) has been both interesting and challenging. It has required a substantial amount of reading in order to assess what is available and what is possible to model. The goal has always been to create a model that is applicable to a real situation, but still general enough to be expanded and analysed. And I personally think the goal is achieved.

It was very interesting to develop an experimental set-up to test the models I created and at the same time learn about microcontroller programming. I would like to give a large thanks to Ingulf Helland and the others at the NTNU instrumentation lab, for all the suggestion and ideas they came up with during this phase of the work. It was too bad that we did not have time to do the actual measurements before the deadline of this work, but hopefully the work will be continued by someone else in the future.

I would like to give a special thanks to my supervisor Prof. Thomas Tybell, for good advice and guidance during this thesis. Thanks to Energy Micro AS for giving me the possibility to work with this interesting problem in such a close to application point of view. I would like to give my co-supervisor at Energy Micro, Alf Petter Syvertsen, special thanks for good guidance and many helpful discussions.

Finally, I would like to give a large thank you to Janne-Lise A. Hegstad for keeping me motivated through all this work, and to my parents for always standing by me, no matter what happens.

Magnus Moreau
Trondheim, June 2013

Contents

List of Figures XIV

List of Tables XV

List of Abbreviations XVII

1 Introduction 1

2 Theory 5

2.1 Definitions and General Principles of Memory Devices 5
2.1.1 Memory Hierarchy 5
2.1.2 RAM Array 6
2.1.3 Feature Size 7
2.1.4 Different Types of Random Access Memories 7
2.2 Challenges in Embedded Memory Design 8
2.2.1 Advantages and Disadvantages of Embedded Memories . . . 9
2.2.2 Embedded Memory Yield 10
2.3 MOSFET Technology and Improvements 11
2.3.1 Fundamentals of MOSFET Devices 11
2.3.2 One Dimensional Electrostatic MOSFET Model 16
2.3.3 Threshold Voltage 16
2.3.4 MOSFET Switching: OFF-State 17
2.3.5 MOSFET Switching: ON-State 21
2.3.6 Important MOSFET Parameters 22
2.3.7 Improving Mobility 24
2.3.8 Scaling of the Channel Length 24
2.3.9 High-k Gate Dielectrics for MOSFETs 26
2.3.10 Different MOSFET Designs for Different Applications 28
2.3.11 Ultimate Scaling of MOSFET, Nanowire FETs 30
2.3.12 Estimating Currents and Scaling Limits for Ultimately Scaled Nanowire FETs 31
2.4 DRAM 33
2.4.1 Basic Operation 33
2.4.2 Main Challenges in DRAM Design 35
2.4.3 Embedded DRAM 36
2.4.4 Ultimate Scaling of DRAM 37
2.5 SRAM 39
2.5.1 Basic Operation 39
2.5.2 Read Static Noise Margin 41

2.5.3 Embedded SRAM . . . 42
2.5.4 Alternative SRAM Cell Designs . . . 42
2.5.5 Impacts of Advanced MOSFET Designs on SRAM Performance . . . 43
2.5.6 Ultimate Scaling of SRAM . . . 44
2.6 FeRAM . . . 44
2.6.1 Basic Operation . . . 44
2.6.2 Main Challenges in FeRAM Design . . . 47
2.6.3 Embedded FeRAM . . . 48
2.6.4 Ultimate Scaling of FeRAM . . . 49
2.7 MRAM and STT-MRAM . . . 50
2.7.1 Magnetic Properties in Layered Structures . . . 50
2.7.2 Field Switching . . . 51
2.7.3 Toggle Field Switching . . . 53
2.7.4 Spin-Torque Transfer (STT) Switching . . . 54
2.7.5 Reading the Cell . . . 55
2.7.6 Embedded MRAM and STT-MRAM . . . 56
2.7.7 Ultimate Scaling of STT-MRAM . . . 57
2.8 PCRAM . . . 59
2.8.1 Properties of Phase Change Materials . . . 60
2.8.2 Basic Operation . . . 60
2.8.3 Multi-Level Storage . . . 62
2.8.4 Embedded PCRAM . . . 65
2.8.5 Ultimate Scaling of PCRAM . . . 67

3 Methodology – Theoretical Estimates 69
3.1 Area Consumption . . . 69
3.2 Estimating the Energy Consumption . . . 70
3.2.1 Power Consumed by the Interconnects . . . 71
3.2.2 Power Consumed to Write or Read One Cell . . . 72
3.2.3 Power Consumed to Open the Access Transistors . . . 72
3.2.4 Total Power Consumption . . . 73
3.3 Estimating Capacitance and Resistance of the Interconnects . . . 73
3.4 DRAM . . . 75
3.4.1 DRAM Refresh . . . 77
3.5 SRAM . . . 77
3.6 FeRAM . . . 79
3.7 MRAM and STT-MRAM . . . 80
3.8 PCRAM . . . 82
3.8.1 Single-Level Cells (SLC) . . . 82
3.8.2 Multi-Level Cells (MLC) . . . 85

3.9 Estimating the Power Associated with the Access Transistors 90
3.9.1 Transistor Design Path 90
3.9.2 Gate Capacitance and Channel Resistance 91
3.9.3 Word Line Voltages 91
3.9.4 Estimating the Gate Width 92
3.10 Further Assumptions 92
3.11 Demands for the Sense Amplifiers 94
3.12 Iso-Feature Size Estimates 95

4 Methodology – Case Studies 99
4.1 Measurement Set-up 99
4.1.1 System Overview 99
4.1.2 Choice of RAM Chips 101
4.1.3 Choice of Microcontroller 102
4.1.4 Demands for the Voltage Regulators 102
4.1.5 Demands on the Printed Circuit Boards Capacitances 104
4.1.6 Microcontroller PCB Set-up 105
4.1.7 RAM PCB Set-up 106
4.2 Microcontroller Software 108
4.2.1 Initializing the Microcontroller 110
4.2.2 Active Period 112
4.3 Estimating the Power Consumption from the Data Sheets 113
4.3.1 Write and Read Energy per Bit 114
4.3.2 Estimating Active Power Consumption 115
4.3.3 Estimating the Passive Power Consumption 116

5 Results and Discussion – Theoretical Estimates 118
5.1 Area Consumption 118
5.2 Capacitance and Resistance of the Interconnects 120
5.3 Cell Write Energy per Bit 122
5.3.1 DRAM 123
5.3.2 SRAM 123
5.3.3 FeRAM 124
5.3.4 MRAM and STT-MRAM 124
5.3.5 PCRAM 125
5.4 Cell Read Energy per Bit 126
5.5 Energy Cost to Switch the Access Transistors 127
5.6 Write Power Consumption 129
5.6.1 DRAM 132
5.6.2 SRAM 132
5.6.3 FeRAM 133

5.6.4 MRAM 133
5.6.5 STT-MRAM 133
5.6.6 PCRAM 134
5.7 Read Power Consumption 134
5.7.1 Relative Signal Strength 137
5.8 Retention Power Consumption 138
5.9 Power vs Duty Cycle 140
5.10 Iso-Feature Size 143
5.10.1 Area 143
5.10.2 Write and Read Power Consumption 144
5.10.3 Power vs Duty Cycle 147
5.11 Important Challenges for the Memory Technologies 148

6 Results and Discussion – Case Studies 151
6.1 Active Power Consumption of the Selected Memory Chips 151
6.2 Passive Power Consumption of the Selected Memory Chips 154
6.2.1 Volatile Memories 154
6.2.2 Non-Volatile Memories 155
6.3 Power vs Duty Cycle 156
6.3.1 Consequences of a Finite Wake up Time 156
6.3.2 Points of Intersection with the SRAM Curves 159

7 Conclusions and Further Work 162

References 165

A Appendix: Matlab Scripts for Theoretical Estimates 179
A.1 Main Script 179
A.2 DRAM 191
A.3 SRAM 193
A.4 FeRAM 195
A.5 MRAM 198
A.6 STT-MRAM 201
A.7 PCRAM 203
A.8 Functions 208

B Appendix: Schematics 210
B.1 MCU Schematics 210
B.2 RAM Schematics 210

C Appendix: C code 220

List of Figures

2.1 Traditional memory hierarchy . . . 6
2.2 Definition of metal half pitch . . . 7
2.3 Schematic of a MOSFET device . . . 13
2.4 Schematic of a MOSFET output characteristics . . . 14
2.5 Gate line edge-roughness and random placed dopants . . . 18
2.6 Standard deviation of the threshold voltage . . . 18
2.7 MOSFET subthreshold slope . . . 19
2.8 MOSFET capacitances . . . 21
2.9 I_D-V_{GS} characteristics . . . 23
2.10 Designs to suppress short channel effects . . . 26
2.11 Schematic of a FinFET transistor. . . . 27
2.12 Optical bandgap vs dielectric constant . . . 28
2.13 Schematic of a single nanowire transistor. . . . 30
2.14 Schematic of a DRAM cell . . . 34
2.15 Gain cell DRAM . . . 37
2.16 $4F^2$ DRAM structure . . . 37
2.17 Pedestal capacitor structure . . . 38
2.18 Schematic of a SRAM cell . . . 40
2.19 SRAM butterfly curves . . . 42
2.20 Schematic of a FeRAM cell . . . 45
2.21 Hysteresis P-V Loop of a Ferroelectric capacitor . . . 46
2.22 Timing diagram for writing a FeRAM cell . . . 46
2.23 Timing diagram for reading a FeRAM cell . . . 47
2.24 Different configurations for the embedded 1T1C FeRAM cell . . . 49
2.25 TMR setup along with the shifted density of states . . . 52
2.26 Field switching magnetic RAM (MRAM) . . . 52
2.27 Toggle switching of the MRAM cell . . . 54
2.28 Schematic of STT-MRAM cell . . . 55
2.29 Equivalent circuit when reading a MRAM or STT-MRAM cell . . . 56
2.30 MRAM cell integrated in the top metal layers of a device. . . . 57
2.31 Schematic of a 1T1R PCRAM cell . . . 61
2.32 Structure of a phase change cell and the temperature vs. time plot . 61
2.33 Cost predictions for different multi-level storage techniques. . . . 62
2.34 Different ways of achieving multi-bit cells . . . 63
2.35 Multilevel storage . . . 64
2.36 Write and verify algorithm . . . 64
2.37 Cross section of a PCRAM embedded in the top metal layers of a device . . . 66

2.38 Cross section of a PCRAM embedded in between the front-end and back-end CMOS process. 66
2.39 Reset current vs equivalent contact diameter for PCRAM devices . 67
3.1 Model for estimating the capacitance and resistance of interconnects. 74
3.2 IV model curve PCRAM. 83
3.3 Conventional MLC PCRAM stair-case-up (SCU) write algorithm . 86
3.4 Flowchart showing the write sequence in a MLC PCRAM 87
3.5 Binary search algorithm for PCRAM read 89
4.1 Block diagram of the measurement system 100
4.2 Schematic of the microcontroller (MCU) PCB 107
4.3 Schematic of PCRAM 109
4.4 Program flow of the C code. 110
5.1 Area consumption 119
5.2 Write energy 122
5.3 Read energy 127
5.4 Energy to switch access transistors 128
5.5 Write power 130
5.6 Breakdown of write power consumption 131
5.7 Read power 135
5.8 Breakdown of read power consumption 136
5.9 Relative signal strength 138
5.10 Power vs duty cycle 141
5.11 Area consumption for $F = 65\,\mathrm{nm}$ 144
5.12 Write and read power for $F = 65\,\mathrm{nm}$ 145
5.13 Breakdown of write and read power consumption for $F = 65\,\mathrm{nm}$ . . 146
5.14 Power vs duty cycle for $F = 65\,\mathrm{nm}$ 147
6.1 Write and read power for the selected memory chips 152
6.2 Normalized power vs duty cycle for selected memory chips A 157
6.3 Normalized power vs duty cycle for selected memory chips B 158
B.1 Schematic of the microcontroller (MCU) PCB 211
B.2 Microcontroller pin-out 212
B.3 Schematic of FeRAM A. 213
B.4 Schematic of FeRAM B. 214
B.5 Schematic of MRAM A. 215
B.6 Schematic of MRAM B. 216
B.7 Schematic of PCRAM. 217
B.8 Schematic of SRAM A. 218
B.9 Schematic of SRAM B. 219

List of Tables

2.1 Memory taxonomy 8
2.2 Mask count adder 9
2.3 Mobility comparisons 24
2.4 Comparison of different MOSFET design paths 29
3.1 Number of bits per µm^2 in 2012. 70
3.2 Number of bits per µm^2 in 2017. 70
3.3 Interconnect resistance parameters 2012. 75
3.4 Interconnect resistance parameters 2017. 76
3.5 Parameters for estimating DRAM cell write energy per bit and the bit line voltage. 76
3.6 Parameters for estimating SRAM cell write energy per bit and the bit line voltages. 79
3.7 Parameters for estimating FeRAM cell write energy per bit. 80
3.8 Parameters for estimating MRAM cell write and read energy per bit and the bit line voltage. 81
3.9 Parameters for estimating STT-MRAM cell write and read energy per bit and the bit line voltage. 82
3.10 Parameters for estimating PCRAM write and read energy per bit and as the bit line voltage. 84
3.11 Access transistor parameters 2012. 93
3.12 Access transistor parameters 2017. 93
3.13 Interconnect resistance parameters for $F = 65\,\mathrm{nm}$. 96
3.14 DRAM parameters for $F = 65\,\mathrm{nm}$ 97
3.15 SRAM parameters for $F = 65\,\mathrm{nm}$ 97
3.16 FeRAM parameters for $F = 65\,\mathrm{nm}$ 97
3.17 MRAM and STT-MRAM parameters for $F = 65\,\mathrm{nm}$. 98
3.18 PCRAM parameters for $F = 65\,\mathrm{nm}$. 98
3.19 Access transistor parameters for $F = 65\,\mathrm{nm}$. 98
4.1 Selected memory chips and main attributes. 102
4.2 Selected microcontroller (MCU) and summary of the main attributes. 103
4.3 Electrical characteristics of the selected memory chips. 103
4.4 Electrical characteristics of the selected microcontroller. 104
4.5 Active current per MHz for the selected memory chips. 105
4.6 Write energy per bit for selected memory chips. 115
4.7 Read energy per bit for selected memory chips. 115
4.8 Parameters for estimating passive power consumption 117
5.1 Capacitance and resistance of interconnects results. 121
6.1 Selected memory chips and main attributes. 151
6.2 Critical passive time for power down. 155

List of Abbreviations

1T1C	One transistor and one capacitor
1T1R	One transistor and one resistor
6T	Six transistors
A/R	Aspect Ratio
BEC	Bottom Electrode Contact
BJT	Bipolar Junction Transistor
BL	Bit Line
CCB	Capacitance Coupled Bit line
CMOS	Complementary Metal Oxide Semiconductor
CNT	Carbon Nanotube
COB	Capacitance Over Bit line
CR	Cell Ratio
CUB	Capacitance Under Bit line
DRAM	Dynamic RAM
EBI	External Bus Interface
ECC	Error Correcting Circuitry
EOT	Equivalent Oxide Thickness
ERD	Emerging Research Devices, a chapter in the ITRS
F	Lithographic Feature size, also called technology node, see figure 2.2b
FeRAM	Ferroelectric RAM
GPIO	General Purpose Input Output
GST	Germanium-Antimony-Tellurium, $Ge_2Sb_2Te_5$
HFRCO	High Frequency RC Oscillator
HFXO	High Frequency Crystal Oscillator
HP	High Performance
I/O	Input/Output

IC	Integrated Circuit
IP	Intellectual Property
ITRS	International Technology Roadmap for Semiconductors
LER	Line-Edge-Roughness
LOP	Low Operating Power
LSB	Least Significant Bit
LSTP	Low Standby Power
MCU	Micro-Controller Unit
MLC	Multi-Level Cell
MOS	Metal Oxide Semiconductor
MRAM	Magnetic RAM
MSB	Most Significant Bit
MTJ	Magnetic Tunnel Junction
PC	Personal Computer
PCB	Printed Circuit Board
PCRAM	Phase Change RAM
PIDS	Process Integration, Devices and Structures, a chapter in the ITRS
PL	Plate Line, sometimes also referred to as the drive line
PZT	Lead Zirconate-Titanate, Pb $(Zr_xTi_{x-1})O_3$
RAM	Random Access Memory
RDF	Random-Dopant Fluctuations
SBT	Strontium Bismuth Tantalate, $SrBi_2Ta_2O_9$
SCE	Short Channel Effects
SCU	Stair-Case Up
SLC	Single-Level Cell
SNM	Static Noise Margin
SOI	Silicon On Insulator

SRAM	Static RAM
STO	Strontium Titanate, $SrTiO_3$
STT	Spin-Torque Transfer
TA-MRAM	Thermally Assisted MRAM
TMR	Tunnel Magneto-Resistance
VCAT	Vertical Channel Access Transistor
WFV	Work-Function Variation
WL	Word Line

1 Introduction

Just a few decades ago computing was only done by large mainframes. This changed in the 1990s, when the computer made its way into the homes of everyday people, with personal computers (PC). In the 2000s and beyond, the embedded area has begun and small computers, such as microcontrollers, have allowed for embedding the computers into everyday appliances, from toothbrushes to sporting equipment. It is not only consumer markets that have embraced this new technology. Embedded computers are also found in harsh industrial environments like the inside of a gas pipe on the bottom of the Nordic sea, or inside the engine of a car [2]. Embedded systems have made it possible to integrate a computer into whatever your mind can think of [3]. So what is an embedded system? Peter Marwedel [3] defined an embedded system as:

> "Embedded systems are information processing systems embedded into enclosing products."

The first recognizably modern embedded system was the Apollo guidance system developed by Charles Dark Draper at the MIT instrumentation laboratory [4]. It was considered to be one of the riskiest parts in the Apollo project, as it used newly developed monolithic integrated circuits(IC).

The reason for this huge development in computers and integrated circuits can be connected to Moore's law. Moore's law states that the amount of transistors on integrated circuits doubles approximately every two years [5]. Consequently, the price of embedded systems has gone down and large improvements in both functionality and processing power have been made. An example on this improvement is the first microprocessor, the Intel 4004, which was designed for calculators and other small systems [6]. This in contrast to today's smart phones supports an enormous amount of features.

The fact that the computers no longer are large static objects has led to many new demands on the performance of the system. One of these new demands come from the fact that many of these new embedded systems are battery operated, and maybe placed in an environment where replacing the battery is not an option [2]. This has led to demands on longer battery life. Battery life is also of importance in hand-held devices, like cellphones, which should at least last for a day before the battery is dead. This is why the semiconductor-industry and -research put a considerable amount of resources in increasing the battery lifetime in these products.

One way to improve the battery lifetime is to make better batteries. Another way is to design the embedded system in such a way that it consumes as little

power as possible. In today's microcontrollers, besides the energy consumed by the processor, the memory is one of the main energy consumers [2]. Traditionally, the chip area of an embedded system was mainly comprised of logical, functional blocks [7]. However, in estimates for future applications up to 90 % of the chip area is expected to be different memory blocks [7, 8].

Conventionally, the memory is divided into two main parts on a microcontroller, the flash memory and the RAM (Random Access Memory). The flash is typically the place where the code for the programs are stored, and is non-volatile, meaning the contents are not lost if the system is powered down. However, the flash has large limitations when it comes to write voltages ($> 10\,\mathrm{V}$), have long write times ($> 1\,\mu\mathrm{s}$) and limited endurance ($\sim 1 \times 10^5$ write cycles) [9]. Therefore, a RAM is used as the working memory of the processor. Conventional RAMs like SRAM (Static RAM) and DRAM (Dynamic RAM), are volatile, so if the power is turned off the stored data is lost.

One simple method of reducing the power consumption of a device and increase battery lifetime is to power down the parts of the device, when they are not in use. This is not an option for conventional, volatile, RAMs, because of the loss of data. This is our motivation for looking at new concepts of RAM technologies, which are non-volatile. These non-volatile RAMs could in principle be turned off when not needed and still retain the data, while still have write voltages of a couple of volt, write times in the order of nano seconds and endurances closing in on DRAMs and SRAMs [9]. Microcontrollers with a conventional volatile RAM, which are inactive, need to spend energy in order to retain the data. If the memory technology is changed to a non-volatile one, it will no longer be needed to spend energy on an inactive memory. This will reduce the total energy consumption of the device.

The memory can either be standalone, on an own chip, or embedded into the chip along with the microprocessor and other circuitry. Standalone chips must be connected to the rest of the device in some way, and this consumes many of the pin-outs from the microcontroller. This again reduces the speed and increases the energy consumption. The reason for this is that the wires from the pins have a parasitic capacitance that has to be charged while transferring data. If the memory is embedded no such pins are needed, and this is not an issue. Still, standalone memories are often more advanced and have larger capacity than an embedded memory. This is due to the reason that the production of an embedded memory has to be incorporated into the production of the rest of the chip [8]. Standalone memories are therefore the driving force in memory technology. However, embedded memories are expected to follow the same trends as standalone, usually with some time lag [10].

In the theory section we will first discuss the basic operation of the different memory technologies, and the physical principles behind them. We will then see if there are any special differences between the embedded and the standalone versions of the memories. If there are any major differences, it is discussed how they affect the device performance.

The ultimate scaling limit is also assessed for each memory technology. When investing in a new technology, it is unpleasant if the first products do not live up to the requirements, but it is not devastating [11]. One can learn from what went wrong and work to improve those aspects. However, if the technology one recently has invested in can not be scaled further than one or two generations, it is a severe problem [11]. Therefore, this work aims to see how the device performance changes when the memory technology in question is scaled further down into the nanometre era. The object is to look for any intrinsic size effects that limits the device performance, and if the energy consumption of the device can be reduced if the technology is scaled down.

In section 3 we will try to estimate the energy consumption of the different emerging and baseline memory technologies. The emerging memories we are looking at are:

- Ferroelectric RAM(FeRAM).
- Magnetic RAM(MRAM).
- Spin-Torque Transfer Magnetic RAM(STT-MRAM).
- Phase Change RAM(PCRAM).

The goal is to estimate the energy in such a way that the different memory technologies can be compared, and we can evaluate which one is the most energy efficient. This is done by focusing on the energy cost to read or write a bit, and the properties of the RAM array. All of the emerging memories are compared with DRAM and SRAM to see how the energy consumption deviates from a conventional memory. This work is looking at how far the technology has come at the present date represented by the year 2012. 2012 is chosen instead of 2013, because this thesis is a continuation of my earlier work [1]. We will also try to estimate what will be available in 2017. 2017 is chosen because it represents five years of time into the future, which is a typical time for developing new technology in the microcontroller industry [2].

The estimates we have done are mainly based on the projections of the international technology roadmap for semiconductors (ITRS). The ITRS is a non-profit organization that publishes a roadmap for the semiconductor-industry every odd

year [12]. This roadmap contains opinions on how different semiconductor technologies will develop in the next fifteen years [12].

We have also done case studies of what is on the market now, in order to see how the devices behave when placed in an end product, as well as comparing these results to our theoretical models. Here FeRAM, MRAM and PCRAM are compared against SRAM. Initially, the plan was to measure the energy consumption of these devices when operated by a microcontroller. However, due to some technical problems, and the fact that a master thesis is only 20 weeks long, there was not enough time to do the actual measurements. Still, a methodology for measuring the power consumption is developed, as will be discussed in section 4.

2 Theory

In this section we will discuss the theory for the different memory technologies. First, we will discuss some general principles and definitions regarding memories and memory systems. Then, we will discuss the overall challenges that separate embedded memories from standalone memories.

We will also discuss MOSFET (Metal Oxide Semiconductor Field Effect Transistor) technology and improvements. MOSFET technology is an important part of a memory system, and are also the main component in Static RAMs (SRAM). A MOSFET transistor is commonly used as a switch in most of the memory technologies, and is the building block in the peripheral circuitry [13]. A discussion of the MOSFET technology will give insight in the possibilities and limitations that this dependence on MOS technology represents.

After we have discussed the MOSFET technology we will look at each of the memory technologies more in detail. First we will look at the operating principle of each technology, then we will look at how each technology can be embedded on a chip, e.g. in a microcontroller. In the final section about each memory we will look at the scaling limits for each technology, to give an estimate of how far it is possible to scale the technology in question.

2.1 Definitions and General Principles of Memory Devices

In this section we will discuss some of the basic principles of memory devices and define some of the main quantities regarding memory systems.

2.1.1 Memory Hierarchy

In memory technology there is a well-known trade-off between device performance (in terms of minimum latency) and memory size (corresponding to the price per bit) [13]. This problem is circumvented by dividing the memory into a hierarchy. The hierarchy ranges from registers, placed close to the processor core, to multiple levels of fast cache memories, then to the working memory known as the RAM (Random Access memory) and finally to various forms of mass storage devices, like hard discs or flash memory. The focus of this work is on RAM technologies. A traditional memory hierarchy is shown in figure 2.1.

It is important to differentiate between the bandwidth and the latency of a system. The bandwidth is defined as the average information throughput per time

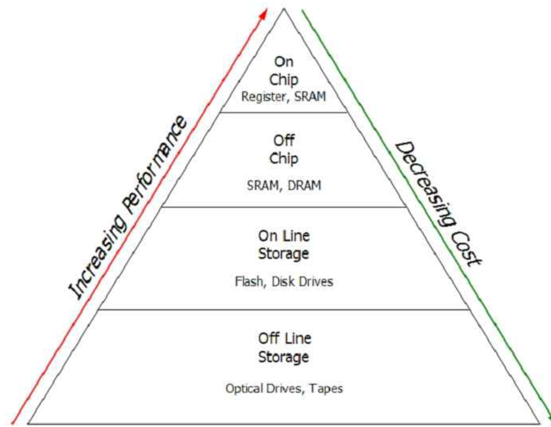


Figure 2.1: Traditional memory hierarchy. From [11].

(Bytes/s), while the latency is the delay time for when the first data can be accessed [13]. An example is hard discs which can provide a high band width (up to 1 Gb/s) if continuous blocks of data are transferred. While they have latencies in the millisecond regime if the data is accessed randomly [13].

2.1.2 RAM Array

To achieve low latencies the RAM stores data in an array, also called a matrix. This allows for fast and parallel access using the lines of these matrices. Usually, the rows of the matrix are called word lines (WL) and the columns are called bit lines (BL).

The matrix in a RAM can either be passive or active [13]. In a passive matrix, the active elements needed are only located at the periphery of the storage matrix [13]. This has the effect that all non-addressed cells in a selected row experience a fraction of the signal, even if they are not located at the addressed node. This raises high demands on the storage mechanics: It must contain two or more clearly distinguishable values for storing data, as well as high quality drivers, so that the signals are reproducible [13]. Furthermore, in resistive based memories parasitic currents can go through unselected cells. These, so called sneak currents, will limit the size of the array. One way to minimize the problem with these currents is to place a two-terminal selector device (e.g. a diode) in series with the memory element. The advantages of a passive array are that the cell size can be made smaller, since no cell area is spent on an active device. It also allows for easier production, and makes it relatively easier to do 3D stacking of cells.

In an active matrix, an active three terminal switch (typically an access transistor)

is placed at each node. This relaxes the criteria on the storage mechanism, because any unwanted signals on un-addresses cell are considerably reduced. The drawback of this approach is that an additional element is needed at each node, and this will limit the storage density [13].

2.1.3 Feature Size

Integrated circuits are characterized after their minimum lithographic feature size, F. Following Moore's law, the feature size of RAMs has decreased by a factor of ~ 0.7 with every generation, as shown in figure 2.2a [10]. The feature size decides the spacing, known as pitch, between the interconnects in the RAM array, as seen in figure 2.2b. Because of this, the densest array that can be made is $4F^2$ [11, 13]. Many of the device properties are tied to the feature size; therefore it is often called the technology node [10].

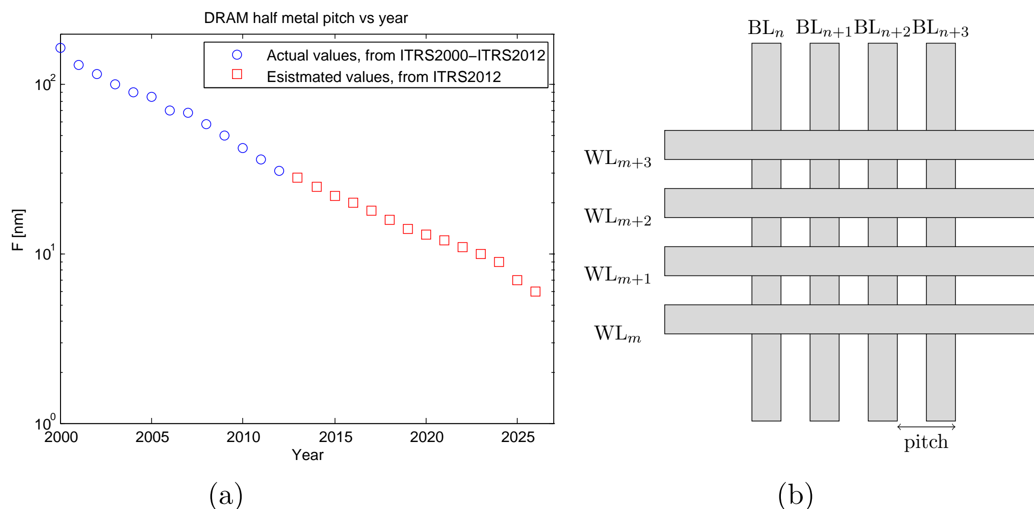


Figure 2.2: (a) Development of DRAM half metal pitch vs year. Adapted from: [14]. (b) Definition of metal half pitch. The figure shows a sketch of a part of a $4F^2$ RAM array and its pitch. The half metal pitch is defined as: F=pitch/2.

2.1.4 Different Types of Random Access Memories

There exists several different types of Random Access Memories, and this report will be describing some of the technologies that are either in production or expected to be available for production in 2017, according to the ITRS. There are two main types of conventional RAM technologies, called SRAM (Static RAM) and DRAM

(Dynamic RAM). They are both volatile, however, while the SRAM is based on interlocked states of electronic gates, the DRAM is based on charge on a capacitor. The emerging memories we are looking at are FeRAM, MRAM, STT-MRAM and PCRAM. The emerging memories are all non-volatile, but the physical mechanisms for storing data are different for some of them, for a brief summary see table 2.1.

Memory technology is often categorized after what kind of circuit elements the cell consists of. In order to save area it is beneficial to use as few circuit elements as possible. However, as the technology is developed it is often necessary to use some additional elements in the beginning in order to improve the stability etc. The DRAM and FeRAM, which consists of a capacitor, C, and an access transistor, T, are labelled 1T1C. The SRAM, which consists of six transistors, is labelled 6T. While the MRAM, STT-MRAM and PCRAM, which are based on a change in resistivity, and contains one transistor and one resistor, are labelled as 1T1R. This is summarized in table 2.1. There exist memory designs that deviate from these classifications, like 9T SRAM, which allows for content addressable memory [15].

	Storage mechanism	Cell elements	Non-volatility
DRAM	Charge on a capacitor	1T1C	Volatile
SRAM	Inter locked state of logic gates	6T	Volatile
FeRAM	Remnant polarization on a ferroelectric capacitor	1T1C	Non-volatile
MRAM	Magnetization of ferromagnetic layer	1T1R	Non-volatile
STT-MRAM	Magnetization of ferromagnetic layer	1T1R	Non-volatile
PCRAM	Reversibly changing amorphous and crystalline phases	1T1R	Non-volatile

Table 2.1: Memory taxonomy. Adapted from: [9]

2.2 Challenges in Embedded Memory Design

In the 1980s the ideal memory for integrated circuits was a standalone memory. It was important for the memory to meet an external I/O standard, and performance and density were not the main concerns [8]. In the 1990s started a new trend, where more and more logic where integrated on the chip moving e.g. the refresh of the DRAM to the chip [8]. Some embedded memories appeared, but

at the time there were a divergence in the memory and logic technology, which hindered the development [8]. From the year 2000, further downscaling of integrated circuits have allowed for larger subsystem sections to be embedded on the chip, and memory is one of them. At the same time, with the increased use of embedded systems in battery operated applications, it has become more important for the total embedded system to be more power efficient; and therefore also the memory.

There are differences in demands for standalone and embedded memories. One reason for this is the wide on-chip buses and parallelism which make high speed operation less essential in order to achieve a high bandwidth. Power can also be reduced by segmenting high capacitance lines and integration of fast I/Os [8].

The most important criterion for embedded memories is comparability with the CMOS(Complementary Metal Oxide Semiconductor) logic process [8]. Specialized memory processes which increase the cost of the logic chip are not accepted unless it comes with large advantages. Examples are planar DRAM cells or gain cell DRAM which does not add process steps to CMOS logic, but do result in larger cell sizes. These designs are preferred due to their lower process costs [8, 16, 17]. Table 2.2 shows a mask adder comparison for the different memory technologies discussed in this work.

	Cell elements	Mask count adder	Source:
DRAM	1T1C	3-6	[18]
SRAM	6T	0-2	[18]
FeRAM	1T1C	2	[19]
MRAM	1T1R	3	[8]
STT-MRAM	1T1R	3	[20]
PCRAM	1T1R	2-4	[21]

Table 2.2: Mask count adder for embedded memories, this is the number of additional masks needed compared to the standard CMOS technology.

2.2.1 Advantages and Disadvantages of Embedded Memories

In this section we will look at some of the main advantages and disadvantages of embedding the memory.

The main arguments for embedding the memory are [8]:

- Improved the performance. The performance is improved as mentioned earlier by increased parallelism, but also by eliminating the need for long inter-

connects in between the chips.

- Reduced power consumption. The power consumption is also reduced by eliminating the need to charge the capacitance of the interconnects from one chip to another. Another way that embedded memories can reduce the power consumption is that it enables the possibility to activate just the amount of memory which is needed at the moment.
- Reduced package cost. The elimination of chip to chip interconnects also reduces the package cost, as it reduces the number of I/O pins needed, and the whole system can be made smaller.

The main disadvantages of using embedded memories in a system are [8]:

- Increase design complexity. This complexity is due to extra processing steps, especially if extra masks are needed. An important part of embedded systems is that the system must be designed for testing, and this will have to be extended to include the memory as well.
- Reduced flexibility. With an embedded memory there will also be less flexibility compared to a standalone variant where the memory can be exchanged more easily.
- Limit the yield. After production, all devices are electrically tested and the amount of accepted devices divided by the total amount of produced devices is known as the yield. When embedding the memory on the chip, it will set limitations on maximum obtainable yield because of the increased amount of parts on a chip which can fail.

Yet despite these disadvantages the pros often out weighs the cons, and embedding the memory is considered a better solution. In many cases the total number of I/Os, and the resulting increased power consumption by the memory, will be dominating the system cost, so that the only solution is to embed the memory [8].

2.2.2 Embedded Memory Yield

Embedding large memories on a chip brings as mentioned large die sizes and reduced yields [8]. This is because embedded memories often are designed with aggressive scaling rules. Therefore they tend to be more prone to manufacturing defects than other cores on the chip [8]. To increase the yield it is common to have some redundancy, i.e. spare elements. It is important to have good design knowledge and failure history in order to produce the adequate number of redundant

elements. However, having the adequate amount of redundant elements is only a part of the solution, it is also important to properly detect defects and allocate the redundant elements in order to solve and repair the defects, in order to improve the yield [8]. In some cases, the repair element can even be from a different memory technology. An example of this is shown by Chun et al. [16], where the embedded DRAM is repaired by replacing defect DRAM cells with SRAM cells placed at the end of each word line.

Traditionally, external testing and repair methods are used to preform memory repair. These external methods rely on extensive use of test equipment and this can contribute to as much as 40 % of the overall manufacturing cost of a semiconductor chip [8]. Therefore, keeping test costs down is essential to lower the total cost of manufacturing.

Today, developing custom intellectual property (IP) cores with built in testing systems reduces the need for external test systems [8]. The IP cores are often reused many times. This reuse of the same process causes an increased knowledge of where failures occur, and allows for optimizing of the failure detection and repair [8].

2.3 MOSFET Technology and Improvements

MOSFET (Metal Oxide Semiconductor Field Effect Transistor) technology is the basic for almost all integrated circuits [22]. In memory technology the MOSFET transistors play an essential role, as they are used both as peripheral circuitry and as access devices [10, 13]. Therefore, development of MOSFET technology is indispensable for almost all memory systems.

The branch of memory technology that benefits most from the development of MOS transistors is the SRAM, as the SRAM cell only consists of transistors, as seen in table 2.1. Because of the importance of MOS technology for memory technology development we will here go into the basic principles of the MOSFET, to understand how the development of the MOSFET technology will impact future memory technologies.

2.3.1 Fundamentals of MOSFET Devices

In this section we will look at some of the fundamental physics behind MOSFET devices and how they work. This will give us the understanding we need in order

to discuss more advanced MOSFET concepts and how they impact the memory technology.

Figure 2.3a shows a cross section of a conventional n-type MOSFET. It consists of two n-doped source and drain regions in a p-type silicon substrate [22]. A gate electrode is placed in between the source and drain regions on top of the active layer in the substrate. The gate electrode is insulated from the substrate by a gate dielectric of thickness d_{ox}. When a voltage V_{GS} is applied to the gate electrode it can attract negative charges in the substrate and form a n-channel from source to drain [23]. This allows a current to flow from source to drain if a voltage V_{DS} is applied to the drain. We will here discuss the operational principles of the MOSFET by looking at the conduction and valence band. In order to keep the discussion as simple as possible, we will here focus on n-type transistors. However, all expressions and results from this discussion can be applied to p-type transistors by reversing the sign of all the voltages and turning conduction band/valence band images upside down [22].

Figure 2.3b shows a schematic of the conduction band for a n-type MOSFET device in the different regions. When a voltage V_{GS} is applied to the gate, the bands at the p-type substrate-dielectric interface is pulled down, and we get a band bending in the z-direction [23]. It is the potential maximum along the channel denoted Φ_{f}^{0}, shown on figure 2.3b with a red circle, that determines the carrier injection and hence the current transport [22]. As we see in figure 2.3, the voltage applied to the gate that affects the potential distribution in the z direction and will lead to a modulation of the current, while a voltage applied to the drain affects the potential distribution along the x-axis and will affect the drift of the current.

In figure 2.4a we see the conduction band profile along the x direction for a fixed gate voltage V_{GS} and increasing bias V_{DS}. The source and drain are so heavily doped that the Fermi level in the source and drain $W_F^{S,D}$ is moved up into the conduction band [22]. If we for the moment neglect scattering, we then have a situation where the carriers injected from the source will occupy states moving from left to right, and the carriers from the drain will give occupy states moving from right to left. The total current can now be found as the difference between the left to right current and the right to left current. The left to right current is given by the product of the charge e, the electron concentration moving left to right $n^{l\to r}$ and drift velocity v [22]:

$$j^{l\to r} = e \cdot n^{l\to r} \cdot v \tag{2.1}$$

while the right to left current is given by the similar product; however, now it is the electron right to left carrier concentration $n^{l\leftarrow r}$ and the velocity is in the

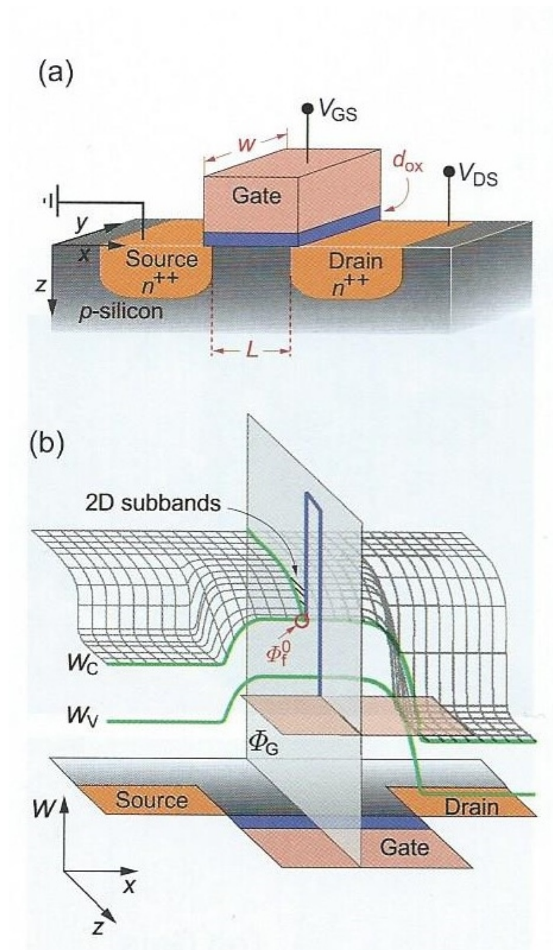


Figure 2.3: (a) A conventional n-type MOSFET where the source and drain contacts are n-doped. The letters in red describe the physical dimensions of the device. (b) A three-dimensional sketch of the conduction band structure of the MOSFET device. From [22].

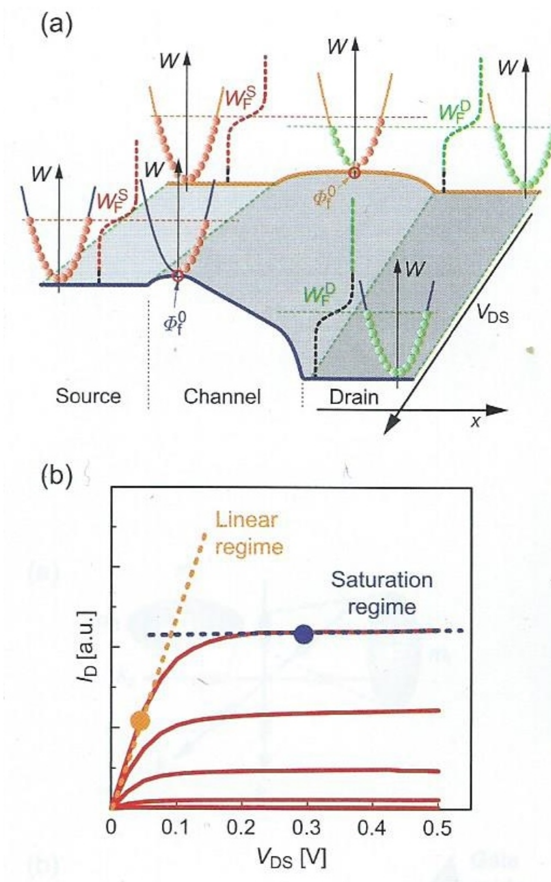


Figure 2.4: (a) Shows the dependence of the conduction band along the x-axis for different bias voltages, the orange line corresponds to situation in the linear regime, while the blue line corresponds to the situation in the saturation regime. (b) The resulting output characteristics. From [22]

negative x direction [22]:

$$j^{l\leftarrow r} = e \cdot n^{l\leftarrow r} \cdot (-v) \tag{2.2}$$

because the current is determined by Φ_f^0, the electron concentration $n^{l\rightarrow r, l\leftarrow r}$ and velocity v only need to be evaluated at this point [22], the electron concentration is given by the integral [24]:

$$n = \int_{W_c}^{\infty} g_C(W) f(W - W_F, T) \, \mathrm{d}W \tag{2.3}$$

where g_C is the electron density of states in the conduction band, W_C is the conduction band energy and $f(W - W_F, T)$ is the Fermi distribution. Generally the velocity v is dependent on the energy W so that we obtain the following expression for the total current [22]:

$$\begin{aligned} j^{tot} = j^{l\rightarrow r} + j^{l\leftarrow r} = q &\int_{\Phi_\mathrm{f}^0}^{\infty} g^{l\rightarrow r}(W) v(W) f(W - W_F^S, T) \, \mathrm{d}W \\ + q &\int_{\Phi_\mathrm{f}^0}^{\infty} g^{l\leftarrow r}(W) \left(-v(W)\right) f(W - W_F^D, T) \, \mathrm{d}W \end{aligned} \tag{2.4}$$

here q is the magnitude of the electron charge. From equation (2.4) we see that the expression within the integral is proportional to the difference in Fermi distributions at the source and drain. This means that if we have a small bias, there is only a small difference in between carriers moving from source to drain and those moving from drain to source, as is seen by the orange line in figure 2.4a. On the orange line at the point Φ_f^0 there is illustrated only a small difference in carriers with positive k value, corresponding to electrons moving in the positive x direction, compared to those with a negative k value and hence moving in the negative x direction. This leads to a linear increase in drain current with increasing bias, as we see as the corresponding orange line in figure 2.4b.

However, when we have a large applied bias, the Fermi level in the drain will lie significantly below Φ_f^0 [22] (blue conduction band profile in figure 2.4a). This means that few of the carriers will have high enough energy to be injected from the drain and into the channel [22]. If we also have that the potential maximum, Φ_f^0, is not influenced by the source drain voltage, V_{DS}, then this will result in a current which stays constant with increased voltage. This is seen as the blue saturation line in figure 2.4b.

2.3.2 One Dimensional Electrostatic MOSFET Model

In order to calculate the current from equation (2.4), we need to find an expression for Φ_{f}^0 as a function of device geometry and applied voltages etc. [22]. In order to this, we need to solve the Poisson equation in three dimensions. However, it is possible to describe the electrostatics by a modified one dimensional model [22]. Using appropriate boundary condition for the surface potential along the interface, $\Phi(x, z = 0) = \Phi_{\mathrm{f}}(x)$, the following equation can be derived [22]:

$$\frac{\mathrm{d}^2\Phi_{\mathrm{f}}}{\mathrm{d}x^2} - \frac{\Phi_{\mathrm{f}} - \Phi_G + \Phi_{\mathrm{bi}}}{\lambda_{ch}^2} = -\frac{e\,(\rho \pm N)}{\epsilon_0 \epsilon_{Si}} \tag{2.5}$$

where Φ_G and Φ_{bi} are the gate potential and built-in potential respectively, ϵ_{Si} and ϵ_{ox} are the relative dielectric constant of the gate substrate and the dielectric, respectively. ρ is the density of mobile carriers and N is a constant charge background due to doping with either donors ('+' sign) or acceptors ('$-$' sign). The length $\lambda_{ch} = \sqrt{\epsilon_{Si}/\epsilon_{ox} \cdot d_{ch} d_{ox}}$ represents the relative length scale of the device, here d_{ox} is the oxide thickness and d_{ch} is the thickness of the source and drain p-n junctions.

That λ_{ch} is the relative length scale can be seen in the limit $\rho \approx 0$ which represents the OFF-state of the device [22]. In this case equation (2.5) can be solved analytically and we get a solution in the form $\Phi_{\mathrm{f}}(x) \propto \exp(-x/\lambda_{ch})$. From this we see that at lengths greater than λ_{ch} the potential differences are smeared out.

2.3.3 Threshold Voltage

When the gate voltage V_{GS} is increased charges is attracted to the gate and can eventually invert a small layer close to the semiconductor-gate interface. This is called inversion [23]. The voltage level which V_{GS} must exceed in order to form an inversion channel under the gate is called the threshold voltage V_{th}. It can be shown that this voltage is given by the following equation [23]:

$$V_{th} = \phi_{MS} - \frac{Q_{it}}{C_{ox}} - \frac{Q_d}{C_{ox}} + 2\phi_F \tag{2.6}$$

here ϕ_{MS} is the difference in work functions of the gate metal and the semiconductor divided by the elementary charge q. Q_{it} represents the effective interface charge per area due to charges trapped at the interface, Q_d represents the charge per area in the depletion region formed in the pn-junction under the gate, and C_{ox} is the capacitance per area of the oxide layer. ϕ_F is the distance between the Fermi level and the intrinsic level in the semiconductor divided by q, and is a measure of

how strongly p-type (n-type for a p-channel MOSFET) the semiconductor channel is.

Variations in the threshold voltage are a large concern for RAM technology especially for SRAM, as it can impact the storage mechanism and reduce the yield [25]. Threshold voltage variations are divided into two types of sources, systematic sources and random sources. Examples of systematic sources are lithography-induced variations in channel length and width, which are deterministic and predictable [25]. Examples of random sources are:

- Random-dopant fluctuations (RDF). Random-dopant fluctuations are due to the randomness in regards to the position of the impurities. This increases when the cell is scaled down. Because, as the gate is scaled down, there is less dopant atoms per device. It can be shown that the threshold fluctuations due to rand RDF is inversely proportional to $\sqrt{W_g L}$, where W_g and L are the gate width and length respectively [25]. One way to reduce the effects of RDF when scaling down the cell is to reduce the channel doping.

- Gate line-edge-roughness (LER). Line-edge-roughness is due to roughness in the lithographically patterned gate electrode because of polymer erosion [25]. Figure 2.5 shows a MOSFET where both LER and RDF effects are present.

- Work-function variation (WFV). Gate work-function variations are because of the randomness of the microcrystalline grains that makes up the gate material, this causes variations in the work function [25] .

These sources are non-deterministic and are expected to be dominant as the cell is scaled down [25]. In figure 2.6 we see that the impacts of these random sources increase with reduced channel length [26].

2.3.4 MOSFET Switching: OFF-State

The off-state of the MOSFET is especially important for the volatile RAM technologies, because this has a direct impact on the static power consumption of the device. In this section we shall see that there is a physical limit to how good the off-state of a MOSFET can be.

The characteristic that defines how good a MOSFET transistor is at switching is the inverse subthreshold slope, S [23]. As we see in figure 2.7, the inverse subthreshold slope is a measure of how fast the current falls off when the gate

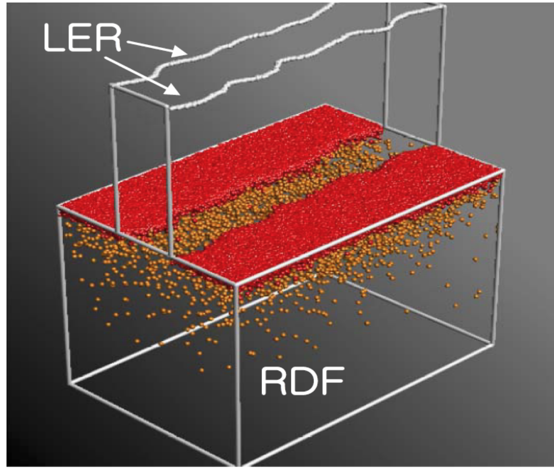


Figure 2.5: A nMOSFET illustrated with gate line edge roughness and random placed dopants. Red coloured atoms are donors, and orange coloured atoms are acceptors. From [25].

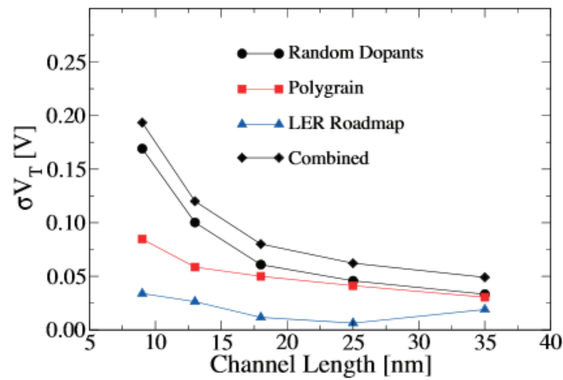


Figure 2.6: Standard deviation of the threshold voltage plotted against channel length for a planar MOSFET. LER estimates are collected from the ITRS roadmap. From [26].

voltage is below threshold. It is defined as [22, 23]:

$$S = \left(\frac{\partial \log(I_D)}{\partial V_{GS}}\right)^{-1} \tag{2.7}$$

Since the inverse subthreshold slope is defined as the reciprocal of derivative of $\log(I_D)$, S should be as small as possible, so that a small change in the gate voltage can cause a significantly change in the drain current [23].

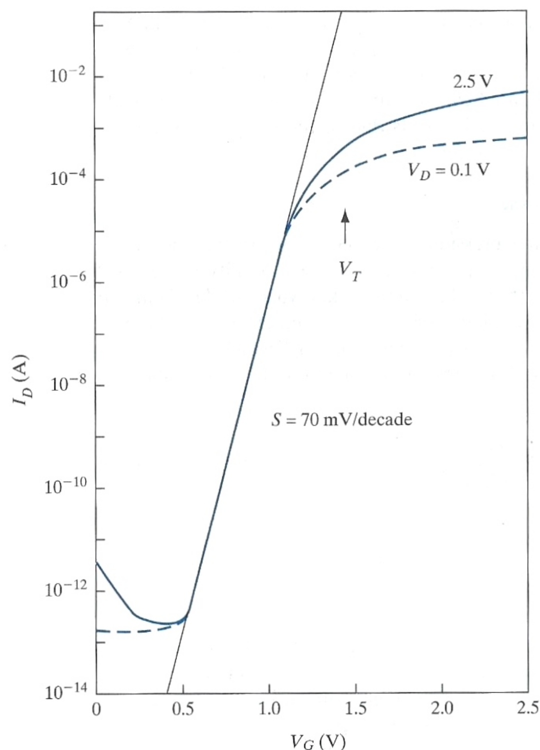


Figure 2.7: MOSFET drain current as a function of gate voltage. We can clearly see that below the threshold voltage the current does not go abruptly to zero, but decreases exponentially with reduced gate voltage. From [23]

In order to quantitatively compute S we use equation (2.4) and employ the Boltzmann approximation for the source/drain Fermi functions, since Φ_{f}^0 lies considerably above W_F^S and W_F^D. It can be shown that the resulting off current is [22]:

$$I_D \propto \exp\left(-\frac{\Phi_{\mathrm{f}}^0 - W_F^S}{k_B T}\right) - \exp\left(-\frac{\Phi_{\mathrm{f}}^0 - \left(W_F^S - qV_{DS}\right)}{k_B T}\right) \tag{2.8}$$

The inverse subthreshold slope can now be calculated as [13]:

$$\begin{aligned} S &= \left(\frac{\partial \log(I_D)}{\partial V_{GS}}\right)^{-1} = \ln(10) \cdot \left(\frac{\partial I_D}{\partial V_{GS}} \frac{1}{I_D}\right)^{-1} \\ &= \ln(10) \cdot \left(\frac{\partial I_D}{\partial \Phi_{\mathrm{f}}^0} \frac{\partial \Phi_{\mathrm{f}}^0}{\partial V_{GS}} \frac{1}{I_D}\right)^{-1} = \ln(10) \cdot \left(\frac{\partial I_D}{\partial \Phi_{\mathrm{f}}^0} \frac{\partial \Phi_{\mathrm{f}}^0}{\partial \Phi_G} \frac{(-q)}{I_D}\right)^{-1} \end{aligned} \tag{2.9}$$

From equation (2.8), we have that $\partial I_D / \partial \Phi_{\mathrm{f}}^0 = -I_D / k_B T$, inserting this gives us:

$$S = \frac{k_B T}{|q|} \ln(10) \cdot \left(\frac{\partial \Phi_{\mathrm{f}}^0}{\partial \Phi_G}\right)^{-1} \tag{2.10}$$

we now see that the missing part, in order to calculate the inverse subthreshold slope, is to find Φ_{f}^0 as a function of the gate potential. This can be done by solving the one dimensional Poisson equation given by equation (2.5). Because the bands are rather flat around Φ_{f}^0, the second derivative term in the Poisson equation can be neglected and we get the relation [22]:

$$\Phi_{\mathrm{f}}^0 - \Phi_G + \Phi_{\mathrm{bi}} = -\frac{q Q_{tot}}{C_{ox}} \tag{2.11}$$

This expression is equivalent to the charge division between the relevant capacitances shown in figure 2.8 [23]. The relevant capacitances are the oxide capacitance C_{ox}, the capacitance due to charges in the depletion region under the channel C_d and the capacitance due to interface charges mainly from unsaturated "dangling bonds" giving a capacitance C_{it}.

As we see in figure 2.8, the capacitance divider can be expressed as:

$$\Phi_{\mathrm{f}}^0 = \frac{C_{ox}}{C_{ox} + C_d + C_{\mathrm{it}}} \Phi_G \tag{2.12}$$

If we now insert this into equation (2.10), we get [22]:

$$\begin{aligned} S &= \frac{k_B T}{|q|} \ln(10) \cdot \left(\frac{\partial \Phi_{\mathrm{f}}^0}{\partial \Phi_G}\right)^{-1} = \frac{k_B T}{|q|} \ln(10) \cdot \left(\frac{C_{ox} + C_d + C_{\mathrm{it}}}{C_{ox}}\right) \\ &= \frac{k_B T}{|q|} \ln(10) \cdot \left(1 + \frac{C_d + C_{\mathrm{it}}}{C_{ox}}\right) \end{aligned} \tag{2.13}$$

from this expression we see that C_{ox} should be much larger than $C_d + C_{\mathrm{it}}$, so that S becomes as small as possible. The depletion capacitance $C_d \propto \sqrt{N_A}$ [22], where N_A is the channel doping concentration. So to achieve a small C_d, the channel doping should be as low as possible. The interface charge should also be reduced in order

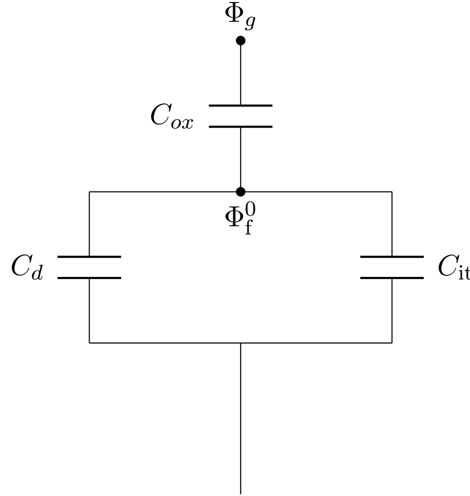


Figure 2.8: The equivalent circuit showing the relevant capacitances for calculating the subthreshold slope.

to lower the interface capacitance. In this case we have $S = k_B T/|q| \cdot \ln(10) \approx 60\,\text{mV/decade}$ at room temperature, and a typical value for a state of the art transistor is $\sim 70\,\text{mV/decade}$ [23]. 60 mV/decade represents a physical limit for any transistor that relies on modulation of carriers injected from a thermally broadened Fermi function. This result is irrelevant of the semiconductor material in use, or the dimensions of the device [22]. This limit is a major obstacle when it comes to reducing the operational voltage, and hence the power consumption of integrated circuits and volatile memory technologies [22].

2.3.5 MOSFET Switching: ON-State

The ON-state of the MOSFET is also important for both logic and memory technologies. An example is when the MOSFET is used as an access device in a RAM, it is then important that this device can deliver enough current for the memory cell to switch its state [10, 13].

We will now derive an expression for the current in the ON-state of the MOSFET. When the gate voltage exceeds the threshold voltage V_{th}, Φ_f^0 is moved close to the source Fermi level and the p-doped semiconductor is inverted at the interface to the gate oxide [22]. This leads to a significant inversion charge density, Q_{inv}, of mobile carriers which is injected into the channel from the source and drain contacts. For a typical MOSFET this charge can be approximated with [22]:

$$Q_{inv} = C_{ox}(V_{GS} - V_{th}) \tag{2.14}$$

If we now use Ohm's law: $J = \sigma E$ with $\sigma = q \cdot n \cdot \mu = (Q_{inv} \cdot \mu)/(W_g \cdot L)$. Where W_g is the width of the device and L is the channel length, and that $E = V_{DS}/L$, we get the following expression for the current in case of a small bias [23]:

$$I_D = \mu \frac{W_g}{L} C_{ox}(V_{GS} - V_{th}) \cdot V_{DS} \tag{2.15}$$

which is the standard text-book result. Here μ is the carrier mobility given by $\mu = q \langle \tau \rangle / m^*$, where $\langle \tau \rangle$ is the mean free time between two scattering events and m^* is the effective carrier mass.

When the applied voltage V_{DS} exceeds $V_{GS} - V_{th}$, there is no voltage difference between drain and gate any more. This means that there is no inversion layer, and hence very few mobile charges. The channel is now said to be "pinched off", which gives a saturation of I_D. In order to satisfy the continuity, the velocity at drain end must be high in order to compensate for the low carrier density. Hence in saturation we get [22]:

$$I_D = \mu \frac{W_g}{L} C_{ox} \frac{(V_{GS} - V_{th})^2}{2} \tag{2.16}$$

This expression is however not valid if the channel is too short. In transistors with a short channel length, the electric fields become correspondingly high, leading to a saturation of the carrier velocity due increased scattering. This scattering mainly happens with optical phonons [22]. In the limit of a very short channel length we get the following expression [22]:

$$I_D^{sat} \approx W_g C_{ox} v_{sat}(V_{GS} - V_{th}) \tag{2.17}$$

This expression is substantially smaller than expected from equation (2.16), it also has no dependence on L and has a linear instead of quadratic dependence on $V_{GS} - V_{th}$.

2.3.6 Important MOSFET Parameters

The most important parameter for a MOSFET is how fast it can discharge or charge another device [22], as this can limit the maximum clock frequency of an IC or RAM.

How fast the MOSFET can discharge another device can be calculated by estimating the device delay time given by [22]:

$$\tau = \frac{C_G W_g L V_{DD}}{I_D} \tag{2.18}$$

where C_G is the total gate capacitance per area and V_{DD} is the supply voltage. Inserting equation (2.16) in case of a long channel MOSFET or equation (2.17) in case of velocity saturation we get [22]:

$$\tau_{\text{long channel}} \propto \frac{L^2}{\mu_{eff}} \cdot \frac{V_{DD}}{(V_{DD} - V_{th})^2} \quad ; \quad \tau_{\text{velocity saturation}} \propto \frac{L}{v_{sat}} \cdot \frac{V_{DD}}{(V_{DD} - V_{th})} \tag{2.19}$$

the device delay time should be as low as possible so that a little charge on the gate can realize a large drain current and allow a high clock frequency [22]. Equation (2.19) suggest several parameters for reducing the delay time, however we also need to take into account the power consumption of the device which is a sum of the dynamic and static power consumption:

$$P = P_{\text{dynamic}} + P_{\text{static}} \tag{2.20}$$

where the dynamic power consumption increases with the square of supply voltage, i.e; $P_{\text{dynamic}} \propto C_G \cdot V_{DD}^2$. The static power consumption increases with the leakage current and supply voltage: $P_{\text{static}} = I_{\text{leak}} V_{DD}$. From this we see that increasing V_{DD} to lower the delay time is not an option as it quadratically increases the dynamic power [22].

Another way that could be used to improve the device performance would be to lower the threshold voltage V_{th}. However, as mentioned earlier, a conventional MOSFET exhibits an inverse subthreshold slope of minimum 60 mV/decade. This means that lowering the threshold voltage would lead to an exponential increase in OFF-state current for zero gate voltage as illustrated in figure 2.9.

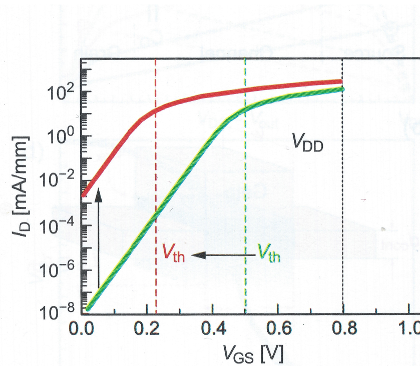


Figure 2.9: I_D-V_{GS} characteristics for two different threshold voltages and constant supply voltage. As we see, lowering the threshold voltage drastically increase the current for $V_{GS} \sim 0$. From [22].

2.3.7 Improving Mobility

Because scaling V_{DD} or V_{th} is not an option, due to the impact on the power dissipation, means that the remaining options for increasing the performance of MOSFET devices are increasing the channel mobility μ or scaling down the channel length L.

Increasing the mobility, $\mu = q \langle \tau \rangle / m^*$, requires changing the material to one with a smaller effective mass, m^*, and/or a reduced rate of carrier scattering, meaning a large $\langle \tau \rangle$ [22]. This can be achieved by changing the substrate material to a semiconductor in the III-V group, as seen in table 2.3. However, this requires a significant change in technology and production lines.

	Energy gap [eV]	Electron Mobility [$cm^2\,V^{-1}\,s^{-1}$]	Hole Mobility [$cm^2\,V^{-1}\,s^{-1}$]
Si	1.1	1350	480
Ge	0.67	3900	1900
GaAs	1.43	8500	400
InP	1.35	4000	100
GaSb	0.7	5000	1000
InAs	0.36	22 600	200
InSb	0.17	1×10^5	1700

Table 2.3: Mobility comparisons of some selected semiconductor materials. All values at 300 K. Adapted from [27].

Another way to increase the mobility, without changing the substrate material, is by applying strain to the silicon. This lifts the six-fold degenerate conduction bands of silicon. This strain therefore reduces the effective mass, m^*, and reduces the amount of valleys the carriers can scatter into, hence increasing $\langle \tau \rangle$. Thus straining silicon is an effective way of increasing device performance without scaling and without changing substrate material. Improvements in mobility up to 100 % are possible with strained silicon [22] .

2.3.8 Scaling of the Channel Length

Scaling down the channel length is another way of increasing MOSFET performance as we see in equation (2.19). Scaling down the channel length does not only increase the performance of the device, but it also allows for integrated circuits with larger complexities or memories with higher densities, due to smaller device dimensions [13, 22]. However, scaling down the device has to be done in an

appropriate manner in order to not lose control of the device, due to short channel effects which appear when the device is heavily scaled. Short channel effects (SCE) will eventually lead to unacceptable leakage currents and can lead to loss of the gate control [22]. It can be shown that SCE is avoided if the following criteria is fulfilled [22]:

$$L \gg \sqrt{\frac{\epsilon_{Si}}{\epsilon_{ox}} d_{ox} d_{cont}} \tag{2.21}$$

where d_{ox} is the thickness of the gate oxide and d_{cont} is the thickness of the contacts. From this we see that ultra shallow contacts and oxide thicknesses are needed in order to suppress SCE. However, as ultra shallow contacts are very hard to fabricate, especially for p-type dopants, due to the rapid diffusion of boron [22]. An alternative method is to use so-called silicon on insulator (SOI). In a SOI structure a thin crystalline silicon layer is placed on top of a buried oxide. In this case equation (2.21) becomes [22]:

$$L \gg \sqrt{\frac{\epsilon_{Si}}{\epsilon_{ox}} d_{ox} d_{SOI}} = \lambda_{ch} \tag{2.22}$$

where d_{SOI} is the thickness of the silicon layer above the buried oxide. From this we see that instead of creating ultra shallow contacts, it is possible to adjust the SOI thickness instead. Note that the square root expression is exactly equal to the effective screening length found from the electrostatic model of the MOSFET [22].

In figure 2.10a we see the conduction band of a MOSFET, where the short channel effects are clearly visible. We see that the source and drain channel p-n junctions overlap and the barrier between source and drain is lowered. This is called drain induced barrier lowering, and results in large leakage currents.

In figure 2.10b we see what happens if the oxide is scaled down, but the contacts are unaltered. This leads to steeper p-n junctions at the substrate because they are close to the metallic gate electrode. However, as we see as the red line in figure 2.10b, the scaling of the gate has a low effect at the potential far away from the interface, so here leakage currents can still flow. In figure 2.10c we see that these leakage currents can be reduced by doping the channel. Such doping gives rise to a depletion capacitance and hence increases the inverse subthreshold slope S, as we stated in equation (2.13). As discussed, channel doping also increases the random variations in threshold voltage due to random-dopant fluctuations (RDF). These effects should be avoided, and doping of the channel is thus not a good solution for ultimately scaled devices. The solution is therefore to scale the contact depth or using silicon on insulator, the short channel effects can then be suppressed without doping the channel as shown in figure 2.10d and e.

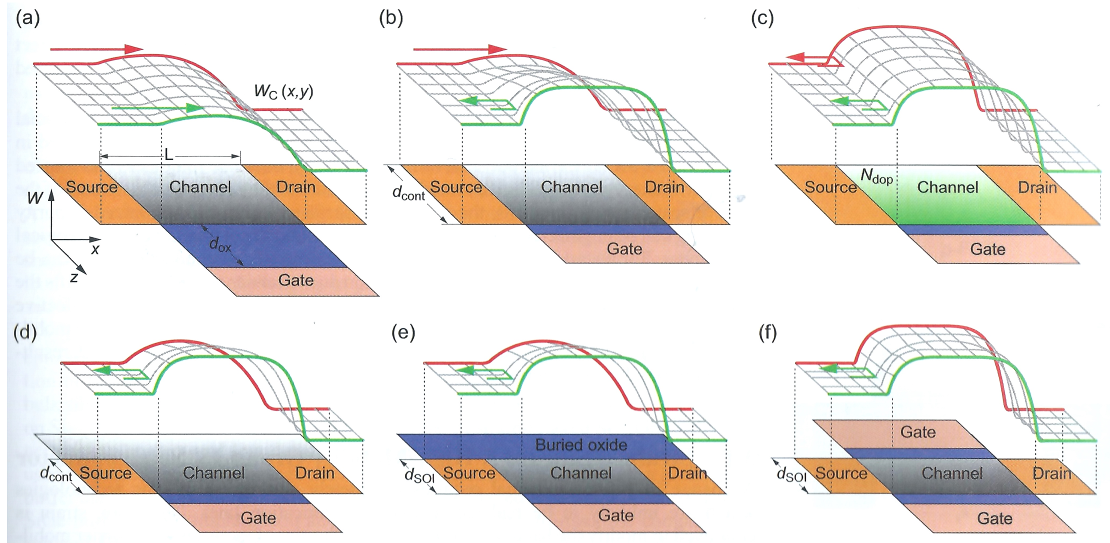


Figure 2.10: (a) Conduction band in MOSFET where short-channel effects are present, as we can see because there is no high barrier between source and drain. (b)-(f) Various architectures in order to suppress short channel effects (SCE). However, as discussed in the text only (d)-(f) represents viable solutions for ultimately scaled devices. From [22].

However, as we see in equation (2.22) both d_{SOI} and d_{ox} have to be as small as possible. Eventually, with continued down scaling this results in threshold voltage fluctuations. This is because of vertical quantization as well as loss of mobility due to roughness scattering [22]. This means that the best control of the channel is achieved with the use of multiple gates. By adding a second gate, as shown in figure 2.10f, we will have, due to symmetry reasons, a device which is equivalent to a single-gate device with a SOI-thickness scaled by a factor of two [22]. Even better control can be achieved by adding a third and fourth gate. Ultimately, the gate all-around structure or nanowire FET is the ultimate MOSFET architecture [22]. An example of a multi-gate structure is the FinFET. The FinFET got its name from the channel, which are placed as a fin going through the gate, as seen in figure 2.11. Intel announced that they will use FinFET for their 22 nm generation processors [28].

2.3.9 High-k Gate Dielectrics for MOSFETs

One of the major advantages for silicon, as a material for highly advanced integrated circuits, is the existence of a natural oxide which fulfils all the properties for a gate dielectric [22]:

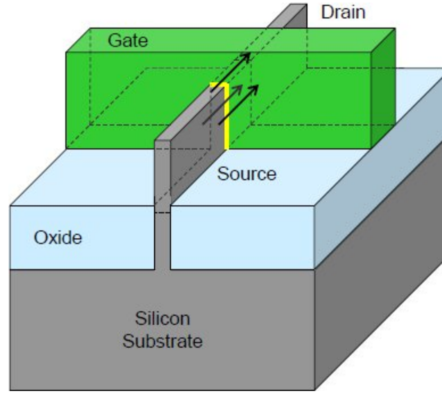


Figure 2.11: Schematic of a FinFET transistor. The channel goes through the gate in a fin-like structure. This will give a tri-gate transistor, because the gate is on three of the channel sides. From [29].

- Excellent insulating properties, with a band gap that exceeds 9 eV.
- Large offsets for both the conduction ΔW_C and valence band ΔW_V, above 3 eV. This effectively suppresses thermal emission of both electrons and holes.
- A saturation of the dangling bonds at the Si-gate dielectric interface, which reduces the interface states and charge. This is important because it reduces the gate interface capacitance in equation (2.13).
- Simple growth of high quality SiO_2 on silicon by thermal oxidation.

However, in order to fulfil equations (2.21) or (2.22) gate dielectrics needs to be scaled down to 1 nm and below to avoid short channel effects [22]. This is a major problem, because at such thin thickness charges can tunnel through the gate barrier. This leakage current increases exponentially with decreasing gate length, this is why exchanging the gate material with a "high-k" material, which exhibits a higher dielectric constant ϵ_k will be beneficial. In order to compare different thicknesses across materials with different dielectric constant, equivalent oxide thickness (EOT) is used; it is defined as [22]:

$$d_{\mathrm{EOT}} = \frac{\epsilon_{r,\mathrm{SiO_2}}}{\epsilon_k} d_{\mathrm{phys}} \quad \text{with } \epsilon_{r,\mathrm{SiO_2}} = 3.9 \tag{2.23}$$

where d_{phys} is the physical gate thickness. As we see materials, with higher dielectric constant will allow for a longer physical gate length, while still fulfilling the demands for suppressing SCE.

Still, there a many challenges in finding a material that is suited for large scale MOSFET integration. Ideally the high-k dielectric should have similar properties

as SiO_2, in order to be a good replacement. It is also important that no interface layers of SiO_x or depletion of the poly silicon is build up, as this will increase the equivalent oxide thickness. If these effects are present it will effectively act as capacitors in series, the total equivalent thickness will in this case be:

$$d_{\text{EOT}} = \frac{\epsilon_{r,\text{SiO}_2}}{\epsilon_k} d_{\text{phys}} + \frac{\epsilon_{r,\text{SiO}_2}}{\epsilon_i} d_i + \frac{\epsilon_{r,\text{SiO}_2}}{\epsilon_{Si}} d_d^{\text{gate}} \tag{2.24}$$

where d_i is the thickness of the interface SiO_x layer, with corresponding dielectric constant ϵ_i and d_d^{gate} is the thickness of the depletion layer with dielectric constant ϵ_{Si}. As the two last terms does not depend on ϵ_k they act as a cut-off for the lowest possible equivalent oxide thickness that is achievable [22].

Furthermore, the high-k material must be compatible with the rest of the CMOS technology and should be thermally stable on silicon [22]. Figure 2.12 shows a selection of dielectric materials that are compatible with silicon [22]. However, as large band offsets also are needed, the choice becomes more limited.

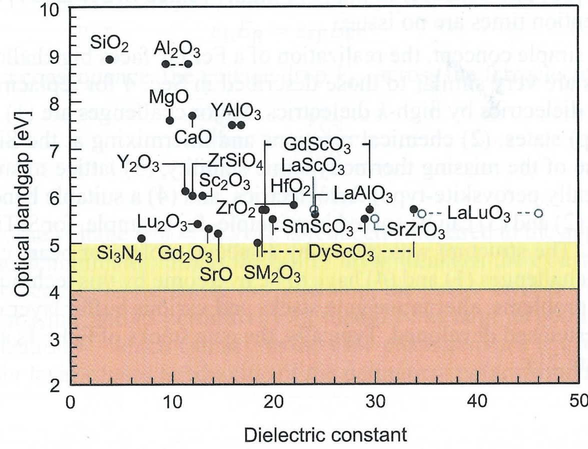


Figure 2.12: Optical bandgap vs dielectric constant of some selected gate dielectrics. In order to provide sufficient insulating properties the bandgap should be above 5 eV. From [22].

2.3.10 Different MOSFET Designs for Different Applications

In the ITRS tables for logic technology, different transistor designs are projected depending on the application. The transistor design is divided into three branches, design for High-Performance (HP), Low Operating Power (LOP) and Low Standby Power (LSTP) [10]. High-performance logic is used in chips where high speed is

of utmost importance and relatively high power consumptions can be tolerated, such as microprocessors for desktop PCs, servers and high performance SRAM caches [18]. Low operating power transistors are typically used for mobile applications where both performance and low power consumption are important. But the battery is assumed to be of high capacity, and the focus is on reduced operating power dissipation. Example applications for LOP-transistors are in laptop computers, and as access and support transistors for non-volatile memory applications [30]. Low standby power transistors are typically for lower-performance consumer type applications, where battery lifetime is of the utmost importance; such as consumer cellphones and volatile memory for microcontrollers [10].

High-performance transistors are designed for the smallest possible delay time, as given by equation (2.19). Hence, the high-performance transistors are the transistors with the most aggressively scaled gate length and lowest threshold voltage. This is why they have the largest leakage currents. These types of transistors are typically only used at the most critical paths, while most transistors on a chip have higher threshold voltage and lower leakage current [10]. The transistors designed for low operating power have the lowest V_{DD}, somewhat lower performance and off-current [10]. The transistors designed for low standby power has the lowest drain source leakage current, as this is the main contribution to the leakage current at room temperature [14].

ITRS assumes that alternate channel materials with higher mobility will be in production by 2018 [14]. The focus for these transistors, when they first comes into production, will be to deliver lower power consumption for the same speed compared to the silicon counterpart [10]. Most likely the material of choice is InGaAs for n-channel devices and Germanium for p-channel.

A comparison between the different design paths is given in table 2.4.

	HP	LOP	LSTP	III-V/Ge
Speed ($1/\tau$)	1	0.5	0.25	1.5
Dynamic Power ($C_G V_{DD}^2$)	1	0.6	1	0.6
Static power (I_{off})	1	5×10^{-2}	1×10^{-4}	1

Table 2.4: Comparison of high-performance (HP), low operating power (LOP), low standby power (LSTP) and high mobility channel (III-V/Ge) MOSFET designs. Adapted from: [14].

2.3.11 Ultimate Scaling of MOSFET, Nanowire FETs

As mentioned earlier, the ultimate gate structure to suppress short channel effects is a gate all around structure, or nanowire FET with a wrap-gate, as shown in figure 2.13. A single nanowire will not deliver enough current, so an array of nanowires is needed [22]. However, in order to understand the physics of nanowires as a switching device it is enough to consider a single device.

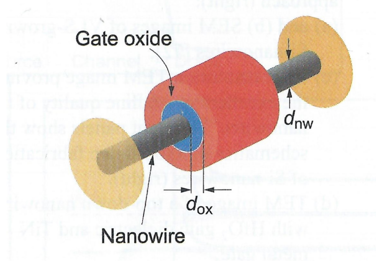


Figure 2.13: Schematic of a single nanowire transistor. From [22].

The gate capacitance for a nanowire FET can be calculated from Maxwell's equations. We start by assuming a total charge Q on the central conductor. If we further assume that the channel length L is much longer than the outer radius of the gate oxide ($L \gg d_{ox} + d_{nw}/2$), we can neglect edge effects. The electric field E in the oxide can then be found from Gauss's law:

$$\begin{aligned} \oint_S \mathbf{E} \cdot \mathrm{d}\mathbf{S} &= \frac{Q}{\epsilon_0 \epsilon_{ox}} \\ E \cdot (2\pi r L) &= \frac{Q}{\epsilon_0 \cdot \epsilon_{ox}} \\ E &= \frac{Q}{L 2\pi \epsilon_0 \epsilon_{ox} r} \end{aligned} \tag{2.25}$$

where r is the distance from the centre of the nanowire. The electric potential between the cylinders can now be found:

$$\begin{aligned} V = V(d_{nw}/2) - V(d_{ox} + d_{nw}/2) &= \int_{d_{ox}+d_{nw}/2}^{d_{nw}/2} -E(r)\,\mathrm{d}r \\ = \frac{-Q}{L2\pi\epsilon_0\epsilon_{ox}} \int_{d_{ox}+d_{nw}/2}^{d_{nw}/2} \frac{1}{r}\,\mathrm{d}r &= \frac{Q}{L2\pi\epsilon_0\epsilon_{ox}} \ln\left(1 + 2\frac{d_{ox}}{d_{nw}}\right) \end{aligned} \tag{2.26}$$

Using the definition of the capacitance $C_{ox} = Q/V$ we finally get:

$$C_{ox} = 2\pi\epsilon_0\epsilon_{ox}\frac{L}{\ln\left(1+2\frac{d_{ox}}{d_{nw}}\right)} \tag{2.27}$$

which again can be shown to give the smallest possible value for the screening length [22]:

$$\lambda = \sqrt{\frac{\epsilon_{nw}d_{nw}^2 \cdot \ln\left(1+2\frac{d_{ox}}{d_{nw}}\right)}{8\epsilon_{ox}}} \tag{2.28}$$

From earlier we have that the gate length should be much longer than the screening length in order to suppress short channel effects ($L \gg \lambda$). With the logarithmic dependence of the screening length on the gate oxide thickness increasing d_{ox} has a smaller impact on λ, and it is therefore easier to scale a nanowire structure compared to a planar structure [22].

2.3.12 Estimating Currents and Scaling Limits for Ultimately Scaled Nanowire FETs

When the nanowire diameter is scaled down, there will eventually be vertical quantization and formation of energetically well separated one-dimensional sub-bands [22]. The one dimensionality will lead to a density of states falling off as $1/\sqrt{W}$. Because the interface charge is proportional to the density of states we will, in bulk MOSFET, have an interface capacitance that increases with gate voltage. However, in a one-dimensional system there will be a spike in the interface capacitance each time a new sub-band is pulled below the Fermi level, and then it will decrease until an new subband is pulled below the Fermi level. This behaviour is called the "quantum capacitance", C_q [22].

If we assume that the nanowire is so thin that only the first subband can contribute, C_q is proportional to $1/\sqrt{W}$. If further we have a very thin gate oxide, the oxide capacitance, given by equation (2.27), can become rather large. If this is the case, the so called quantum capacitance limit can be reached where $C_{ox} \gg C_q$. In this limit it can be shown that $\partial\Phi_f^0/\partial\Phi_D \to 0$ [22], and from equation (2.12) we have $\partial\Phi_f^0/\partial\Phi_G \to 1$. This means that the gate has perfectly control over the channel, in both the on- and off-state.

When the channel is scaled down it will eventually be much shorter than the mean free path between scattering events. In other words we will have ballistic transport through the channel. We can now find the current through equation (2.4), which

now can be written as [22]:

$$I_D = 2q \int_{\Phi_f^0}^{\infty} g^{1D}(W)v(W)\left(f_S(W - W_F^S, T) - f_D(W - W_F^D, T)\right) \mathrm{d}W \tag{2.29}$$

where $g^{1D}(W)$ is the one-dimensional density of states. We have that $g^{1D}(W)v(W) = 1/h$ where h is the Planck constant, and because $\partial\Phi_f^0/\partial\Phi_G \to 1 \implies \Phi_f^0 = \Phi_G + const.$, the integral can be solved analytically. With a large applied bias the drain Fermi function can be neglected, and hence the current is proportional to the integral of the source Fermi function. With a small applied bias the difference of the fermi functions can be expanded as a Taylor series. Doing this, it can be shown that the drain current as a function of V_{DS} and V_{GS} is given by [22]:

$$I_{D,\text{low bias}} = \frac{2q^2}{h} V_{DS} \frac{1}{\exp\left(\frac{-qV_{GS}+const.-W_F^S}{k_B T}\right) + 1} \tag{2.30}$$

$$I_{D,\text{sat}} = \frac{2qk_B T}{h} \ln\left[\exp\left(\frac{W_F^S - (-qV_{GS} + const.)}{k_B T}\right) + 1\right] \tag{2.31}$$

which is an interesting result, because the current does no longer depend on device geometry or material properties. The conditions to achieve a device like this are: One-dimensionality, quantum capacitance limit and ballistic transport [22].

For the ultimate scaled nanowire FET, the limiting factor will become direct source-to-drain tunnelling [22]. Up till now we have only considered carriers emitted over the potential barrier in the channel, but when the channel is short enough, the carriers can tunnel directly from source to drain.

We start by considering the leakage current for carriers thermally excited over the barrier. If we assume small V_{DS} and zero gate voltage, the channel has a band gap, ΔW_g, and effective mass, m^*. The current be found from equation (2.30) and will be [22]:

$$I_D^{\text{thermal}} = \frac{2q^2}{h} V_{DS} \frac{1}{\exp\left(\frac{-\Delta W_g}{k_B T}\right) + 1} \tag{2.32}$$

To find the leakage current from direct source to drain tunnelling we assume a step-function-like potential in between source and drain, the tunnelling current is then approximately proportional to [22]:

$$I_D^{\text{tunnel}} \propto \frac{2q^2}{h} V_{DS} \cdot \exp\left(-\frac{2L}{\hbar}\sqrt{2m^*\Delta W_g}\right) \tag{2.33}$$

In order to make sure that the device provides ideal switching as a conventional MOSFET the tunnelling current should be less than the thermal current (i.e. $I_D^{\text{tunnel}} < I_D^{\text{thermal}}$). As we see, this sets limits on the gate length L. For silicon this limit has been shown to be at a gate length of approximately 6 nm [31]. However, as can be seen from this simplistic model, other materials with smaller effective mass, m^*, will have increased tunnelling current and will thus reach this limit at longer gate lengths [22]. The ultimate scaling limits of MOSFETs can be a limiting factor for memory technologies in cases where the access device is the limiting factor, or in SRAM applications where the cell itself are made with MOSFETs.

2.4 DRAM

In this section we will discuss the Dynamic RAM (DRAM) with a focus on the aspects of embedded DRAM and the ultimate scaling limits for the DRAM cell. However, we will start by repeating the basic operation which were also presented in section 2.4 of my earlier work [1]. This is done for completeness, and to get a good understanding of how the different aspects of DRAM design affects the operation.

The Dymamic RAM was patented in 1967 [32], and introduced to the market by Intel Corporation in 1972 [13]. The simple structure, where data is stored as charge on a capacitor, has made it possible for DRAM to be one of the leading RAM technologies for several decades [13]. It is called dynamic because the stored charge constantly fades away, even with applied power. So the cells must be continuously read and refreshed with a given time period [15], in order retain information.

2.4.1 Basic Operation

Different cell structures have been developed over the years, but the most widely used is the simple structure of a 1T1C DRAM cell, which is shown in figure 2.14. This has been the cell structure of choice because of the very few circuit elements in each cell, leading to a small cell size. As described in my earlier work [1], the access transistor, labelled T in the figure, acts as a switch for the capacitor C_{cell}. A logic one is stored as a charge on the capacitor while a zero is stored as the absence of charge on the capacitor.

When **writing** to the cell, one starts making the channel conducting by applying a voltage $V_{DD} + V_{th}$ to the gate of the access transistor, through the word line. This voltage is known as a boosted V_{DD} and is applied to get enough drive current

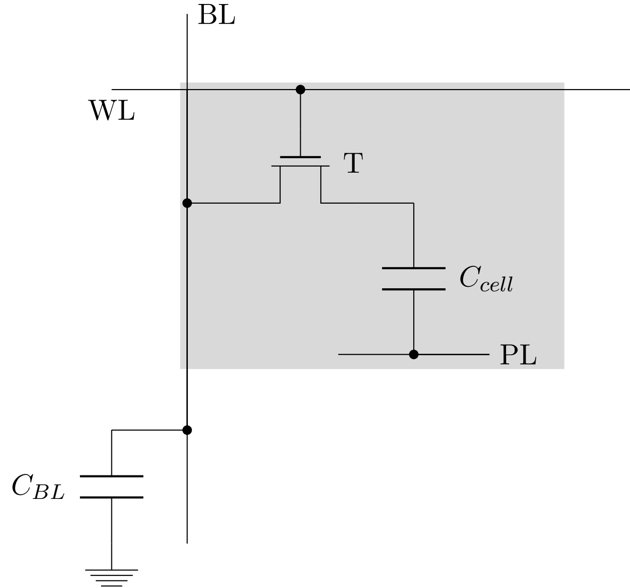


Figure 2.14: Schematic of a 1T1C DRAM cell: the cell contains an access transistor, which can be used as a switch, and one capacitor for storing the information. The cell can be accessed through the bit line and word line.

through the access transistor which often has an increased threshold voltage in order to reduce leakages [14]. While the access device is conducting, a voltage $+V_{DD}/2$ is applied on the bit line for writing one, and $-V_{DD}/2$ is applied on the bit line for writing a zero. The plate line, labelled PL, is kept at $+V_{DD}/2$ [13]. When the writing process is done, the capacitor is isolated by opening the transistor switch.

In order to **read** the cell, the charge must be sensed with an external circuit, often called a sense amplifier. The reading starts with closing the transistor switch, which leads to a redistribution of the stored charge between the cell capacitance C_{cell} and the capacitance of the bit line C_{BL}. This leads to a voltage change on the bit line, depending on whether the capacitor was charged to $+V_{DD}/2$ (for one) or $-V_{DD}/2$ (for zero) [13]:

$$V_{BL} = \left(1 \pm \frac{C_{cell}}{C_{cell} + C_{BL}}\right) \frac{V_{DD}}{2} \qquad \begin{matrix} + \text{ for one} \\ - \text{ for zero} \end{matrix} \tag{2.34}$$

Since the readout removes the charge from the capacitor in order to sense it, a read operation is destructive. Therefore, a read operation has to be followed by an immediate write-back so that the contents not are lost.

The contents of the DRAM are volatile, in other words, the charge on the capacitor

is lost if the power supply is cut. In addition, the stored charge decreases with time due to two different phenomena [13]:

- Leakage current through both the transistor and capacitor due to non-zero conductance.
- Dielectric losses which can be characterized as dielectric relaxation currents.

Because of this, DRAM requires to be refreshed periodically. The stored charge should not decrease by more than 10 % within one refresh period. The refresh is usually done for all the word lines at a time.

2.4.2 Main Challenges in DRAM Design

The main challenges in DRAM design are to keep the capacitance of the cell almost constant while scaling down the cell. This is needed in order to compensate for the leakages and to get a large enough signal for the sense amplifier to detect [33]. The common criteria for the cell capacitance are that it should be above 20 fF [13]. From the formula of a parallel plate capacitor we can get insight in how to maintain such a high capacitance value while scaling down the cell:

$$C_{cell} = \epsilon_0 \epsilon_{r,\text{eff}} \frac{A_S}{t_{\text{phys}}} = \epsilon_0 \epsilon_{r,\text{SiO}_2} \frac{A_S}{t_{\text{eq}}} \tag{2.35}$$

here A_S is the total area of the capacitor, t_{phys} is the physical thickness of the dielectric, $\epsilon_{r,\text{eff}}$ is the effective relative permittivity and ϵ_0 is the vacuum permittivity. The equivalent thickness t_{eq} is defined as:

$$t_{\text{eq}} = \frac{\epsilon_{r,\text{SiO}_2}}{\epsilon_{r,\text{eff}}} t_{\text{phys}} \quad \text{with } \epsilon_{r,\text{SiO}_2} = 3.9 \tag{2.36}$$

note that this is equivalent to the definition of equivalent oxide thickness for MOSFET devices as stated in equation (2.23). If we now look at equation (2.35), we see that while scaling down the cell dimensions, the capacitor area decreases. To counteract this effect there are two main ways of fulfilling the capacitance, criteria of at least 20 fF, when reducing the footprint area: Using materials with a higher dielectric constant ϵ_r (often referred to as high-k materials) and increasing the area through 3D-stacking [13]. Reducing the physical thickness is not a feasible option for state of the art DRAM capacitors, due to exponentially increased leakage from tunnelling currents [13]. For a discussion of challenges in 3D stacking and the use of high k-materials in DRAM capacitors please see section 2.4 of my earlier work [1].

2.4.3 Embedded DRAM

SRAM has been the dominant choice for embedded memories for a long time. However, in recent years, especially for multi-core processors, embedded DRAMs are becoming increasingly popular, due to their reduced area and lower power consumption [34,35]. For embedded DRAM, different demands have caused a separation in technologies for standalone and embedded technologies since the 1990s [33]. As mentioned in the challenges for embedded technologies, the embedded memories can often take a penalty in area consumption in exchange for a simpler and faster cell. Faster cells are a high priority, as the embedded DRAM are mainly replacing embedded SRAM in high performance applications [34].

Where the standalone DRAMs uses long word lines and bit lines in order to be as area effective as possible, the embedded DRAM often has smaller sub arrays with shorter interconnects. Because this reduces the bit line capacitance, it allows for the embedded DRAM to have a lower cell capacitance, typically 5 fF [33]. As seen in equation (2.34), when the bit line capacitance is reduced, the same signal magnitude can be achieved with a smaller cell capacitance. Also due to the smaller capacitances, and generally due to less resistance in the larger capacitor plates, the embedded DRAM can work at higher clock frequency because of smaller RC delays [9,33]. A drawback with the reduced cell capacitance is the reduced retention time, typically ~ 4 ms compared to ~ 64 ms for standalone DRAMs [9]. Increasing the refresh period is critical from a low power perspective as this will drastically reduce the passive power consumption [35]. However, when the capacitance is reduced the energy cost to refresh a cell is also lowered [9].

The embedded DRAM technology prefers simpler processing and simpler capacitor structures than standalone, due to cost issues, as discussed in section 2.2. Because of this, the area of standalone DRAM cells are 4-6F^2, while the embedded 1T1C DRAM varies from 12-50F^2 depending on application and process [9]. This high focus on a simpler processing is taken even further with gain cell DRAMs. Gain cell DRAMS consists of multiple transistors (typically 2-4), but without an explicit transistor [35], as illustrated in figure 2.15. Instead of using an explicit capacitor the charge is stored in the capacitors of the MOS gates. This capacitance is typically low (< 1 fF [35]), but because of the multiple transistors the cell has a gain which amplifies the signal [33]. This allows for low voltage operations, and because the cell only consists of transistors it can be produced with no additional masks [16]. However, since the capacitance is low, the retention times are in the order of 100 µs [16]. Despite these issues, the gain cells DRAMs consumes 25 % of the static power compared to a SRAM produced with similar technology [16].

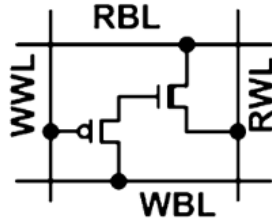


Figure 2.15: Example of gain cell DRAM for embedded applications. The cell has two transistors, but with no explicit capacitor. RBL and WBL is the read and write bit line respectively, while RWL and WWL are the read and write word lines respectively. This example has an area consumption of $65F^2$. From [35].

2.4.4 Ultimate Scaling of DRAM

In this section we will look at what are the physical limits for the DRAM cell. The ultimate limit for any memory device, when it comes to array efficiency, is a $4F^2$ structure [13, 36]. The DRAM cell has benefited greatly from the development of multi-gate transistors the later years, in order to keep the leakage currents low with further scaling [10]. For DRAM to achieve the $4F^2$ limit it requires the use of a vertical channel access transistor (VCAT) and a pedestal capacitor structure, as discussed in section 2.4.3 of my earlier work [1]. As we see in figure 2.16, the VCAT is a gate all around MOSFET, where the gate is wrapped around the vertical channel.

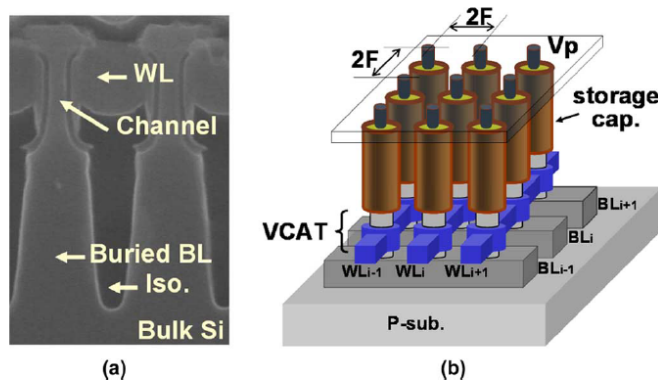


Figure 2.16: (a) Cross section of $4F^2$ DRAM structure. (b) Schematic of vertical channel access transistor (VCAT) and pedestal capacitor structure needed for $4F^2$ DRAM. From [37].

The bottom electrode post of the pedestal capacitor can not have a diameter that is less than the feature size, F [13]. In this case the ultimate cell capacitance,

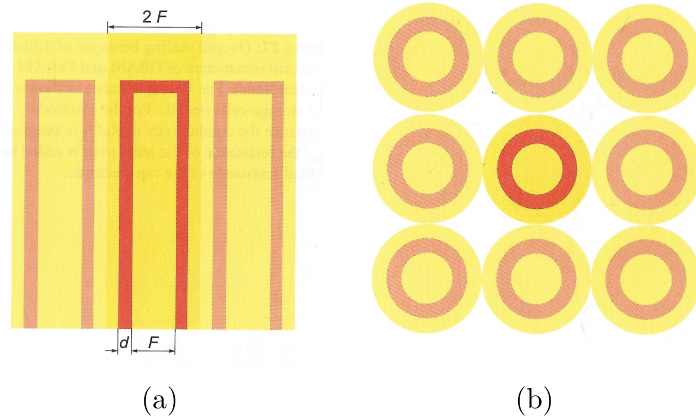


Figure 2.17: Schematic of pedestal capacitor for DRAM. (a) Side view, transistors below not shown. (b) Top view. From: [13].

given by equation (2.35), can be estimated by [13]:

$$C_{cell} = \epsilon_0 \epsilon_{r,\max} \frac{A_S}{t_{\min}} \approx \epsilon_0 \epsilon_{r,\max} \frac{\pi \left(A/R + \frac{1}{4} \right) F^2}{t_{\min}} \tag{2.37}$$

where $\epsilon_{r,\max}$ is the largest achievable dielectric constant, $t_{\min}$ denotes the minimal thickness of the capacitor and A/R is the aspect ratio of the capacitor. The minimal thickness was shown in [38] to be about 5 nm. For thicknesses smaller than 5 nm, leakage currents due to tunnelling through the dielectric barrier become too large for the cell to fulfil the charge loss requirement. The maximum dielectric constant at this thickness is 200 for polycrystalline strontium titanate, $SrTiO_3$ (STO) [13], this gives an equivalent thickness of 0.1 nm as the ultimate scaling limit. Keeping the demand of a cell capacitance of at least 20 fF for standalone DRAM, the aspect ratio can now be calculated as a function of feature size F.

With a minimum physical thickness of the capacitor dielectric of 5 nm, we see that the ultimate scaling limit for the pedestal capacitor structure, shown in figure 2.17, is for a feature size $F = 10$ nm. This is because of when the feature size approaches 10 nm, the thickness of the top (outer) electrode goes to zero. This will again lead to a contact resistance of the top (outer) electrode which approaches infinity [13]. This will be a scaling limit due to the fact that the charging time of the capacitor approximately is given by $\tau = RC$ [13]. With $F = 10$ nm there is no room for a top (outer) electrode, therefore the limit for a practical DRAM is typically set at 12 nm [13]. With dedicated etching techniques it might be possible to structure the bottom (inner) electrode smaller than the feature size, F. If that is the case, more space is available for the outer electrode and it might be possible to scale the cell down to a slightly lower feature size [13].

2.5 SRAM

In this section we will discuss the Static RAM (SRAM). We will start by looking at the working principle and the main challenges in SRAM design. Finally we will look at the ultimate scaling for SRAM. The SRAM uses a static latch made by two cross-coupled inverters to store information. And the development of the SRAM is therefore governed by the development of the MOS process as the cell only consists of transistors. One example is the SRAM speed, which has been enhanced by the scaling of the gate length as can be seen in equation (2.19).

2.5.1 Basic Operation

A schematic of a standard SRAM cell is shown in figure 2.18. It shows that the SRAM cell consist of two cross-coupled inverters, with two access transistors or pass-gate transistors. The pMOS transistors, T_3 and T_5, works as pull-up transistors, while the nMOS transistors, T_4 and T_6, works as pull-down transistors. This circuit has two stable states designated as one and zero, and with this set-up a total count of six transistors is needed per cell.

A **read operation** is performed by setting the two bit lines high, and selecting the word line. As a result, the side storing a logical zero (i.e. a low voltage) will discharge the high voltage though the access transistor and the nMOS pull-down transistor. This will cause a differential voltage to develop between the two bit lines [25]. This differential voltage should be large enough for a sense amplifier to detect the state of the cell. However, the differential voltage should not be too large, because this could cause the channel of the other nMOS pull-down transistor to start conducting, and flip the state (i.e. if the differential voltage is too large the read is no longer non-destructive) [39]. The β-ratio, also called the cell ratio, should be large enough such that a read disturbance like this does not occur [25]. For a symmetric device the cell ratio CR is defined as [40]:

$$CR = \frac{W_{g,4}/L_{g,4}}{W_{g,1}/L_{g,1}} = \frac{W_{g,6}/L_{g,6}}{W_{g,2}/L_{g,2}} \tag{2.38}$$

where $W_{g,i}$ and $L_{g,i}$ is the gate width and length respectively of the corresponding transistor in figure 2.18. In the case where all gate lengths are equal, equation (2.38) can be simplified as:

$$CR = \frac{W_{g,4}}{W_{g,1}} = \frac{W_{g,6}}{W_{g,2}} \tag{2.39}$$

For a **write operation**, a write driver circuit drives the bit lines to complementary voltage level before the word line is selected [25]. On the side which is logical

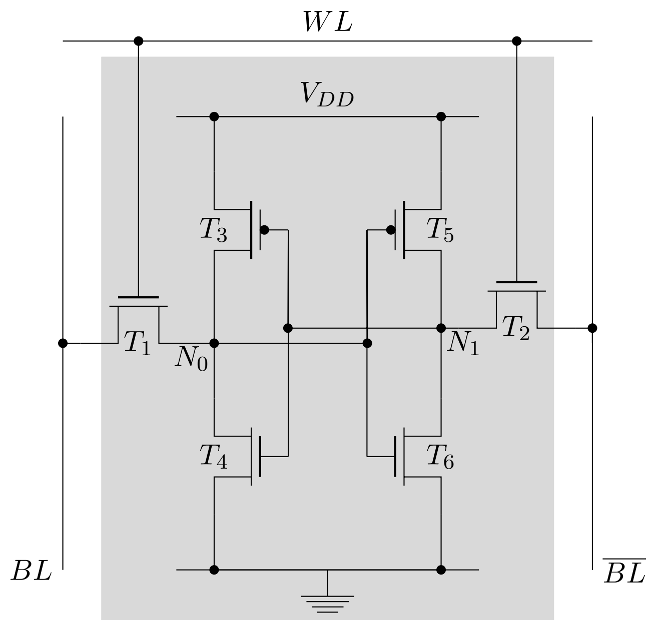


Figure 2.18: Schematic of a SRAM cell. The cell consists of 6 transistors two works as access transistors, and the other four is coupled as two cross-coupled inverters. The pMOS transistors, T_3 and T_5, works as pull-up transistors while the nMOS transistors, T_4 and T_6, works as pull-down transistors.

zero (i.e. a low voltage), the internal voltage is discharged through the access transistor. The inverters then raise the voltage on the opposite side and latch the cell [25]. It is now important that the discharging strength of the pass gate transistor overcomes the restoring strength of the pull-up transistor. Pull up ratio PR, should be sufficiently small to ensure that a write failure does not occur. The pull-up ratio for a symmetric SRAM cell is defined as [40]:

$$PR = \frac{W_{g,3}/L_{g,3}}{W_{g,1}/L_{g,1}} = \frac{W_{g,5}/L_{g,5}}{W_{g,2}/L_{g,2}} \tag{2.40}$$

similar to the cell ratio, if the gate lengths are all equal, equation (2.40) can be simplified as:

$$PR = \frac{W_{g,3}}{W_{g,1}} = \frac{W_{g,5}}{W_{g,2}} \tag{2.41}$$

As we see, it is not possible to get a compact cell, large cell ratio and small pull-up ratio at the same time. Therefore, in high density 6T SRAM it can be shown that a good trade-off between read-ability and write-ability is achieved when CR and PR are 2 and 1 respectively [40]. However, these numbers should be optimized for each design, in order to obtain maximum yield [25].

The logic state stored in the cross-coupled inverters will retain as long as a power supply is present [15]. However, if the power supply is turned off the information is lost. To reduce the power spent on retaining data, transistors that are designed especially for low leakage currents are needed as discussed in the MOSFET chapter. As discussed, the cell is more prone to failure while either writing or reading. This means that lower margins are needed when only retaining data. By using this approach it makes it possible to reduce the power consumption by reducing the supply voltage, if it is known that the device should be inactive over a longer period [25].

2.5.2 Read Static Noise Margin

Dealing with read stability failures is one of the biggest challenges for SRAM design [41], and the most common metric for SRAM read stability is the read static noise margin (SNM) [25]. SNM is a measure of how much static noise can be tolerated before a read failure occurs. The static noise margin can be found by drawing and mirroring the inverter characteristics, as shown in figure 2.19 [42], these curves are often called butterfly-curves [25]. The next step is to find the largest squares (squares C and D) between the two inverter characteristics (line A and line B) in figure 2.19. The read static noise margin (SNM) is now defined as the diagonal of the smallest square [43].

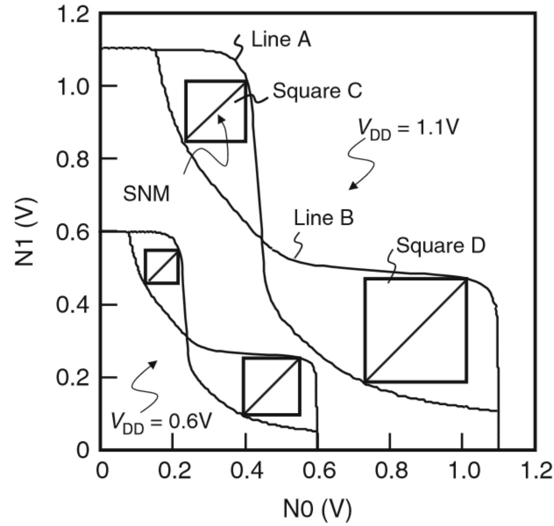


Figure 2.19: So called butterfly curves and the corresponding read static noise margin (SNM) for the SRAM cell, for $V_{DD} = 1.1\,\text{V}$ and $V_{DD} = 0.6\,\text{V}$. From [43].

As is also seen in figure 2.19, the static noise margin is reduced when the supply voltage V_{DD} is scaled down. As discussed in the MOSFET chapter, the dynamic power is proportional to the square of the supply voltage. This means that if the supply voltage is scaled down in order to reduce the power consumption, the read static noise margin is also reduced, and the device is more prone to failures.

2.5.3 Embedded SRAM

The SRAM is the most common memory in embedded applications [8]. The reason for this is that SRAM cells can be made without any added masks, as seen in table 2.2 on page 9. The SRAM is also fully compatible with the CMOS technology and allows for seamless integration with logic circuits [44]. The embedded SRAM has also been the most used technology in the parts of the memory hierarchy that is closest to the processor. This is because embedded SRAM has provided the highest random access speeds [44].

2.5.4 Alternative SRAM Cell Designs

As mentioned earlier, it is important to tune the widths of the transistors 6T-cell. This is needed to fulfil both the read and write margin specification to ensure proper operation, there is therefore a fundamental trade-off between cell yield and

cell area [25]. This situation gets even worse by the need to include margin for the process induced variations in threshold voltage, V_{th}, as discussed in the MOSFET chapter.

To solve these issues alternative SRAM cell designs with 8 to 10 transistors are investigated. These can increase the static noise margin considerably by providing write and read buffers. This removes the problem with the conflicting demands on access transistor for writing and reading [40]. However, as these designs consist of more elements in each cell, they generally consume more area. Still these designs can be the solution to overcome worsened read and write margins with further down scaling [40].

2.5.5 Impacts of Advanced MOSFET Designs on SRAM Performance

Silicon on insulator and multi-gate designs has been shown to improve SRAM performance [25]. One of the main reasons for this is, that as discussed in the MOSFET section, SOI and multi structures do not need channel doping to reduce short channel effects. Channel doping increases the inverse subthreshold slope and threshold voltage variations due to random dopant fluctuations [10], so mitigating these problems has large impacts on SRAM performance.

Due to these differences it was shown in [25] that 22 nm gate length silicon on insulator designs can achieve a six standard deviation (6σ) yield with approximately 25 % area savings compared to a cell consisting of 22 nm gate length planar bulk transistors designed for similar off-state current and yield. The silicon on insulator design can also operate at a lower supply voltage, because it provides higher drive current and reduced variability [25].

Multi-gate structures like the FinFET are assumed to continue this trend. However, one challenge for SRAM designs with FinFETs is that it is not possible to exactly tune the effective width of each transistor to increase the current. It is only possible to tune the width by using additional fins, which leads to a quantization of the current [28]. This quantization makes it harder to adjust the pull-up and cell ratios in order to maximize the yield [25]. Due to these issues, and many manufacturing difficulties, it is anticipated that the FinFETs or other multi-gate transistor structures will initially only be used in SRAM arrays where low leakage is imperative [25].

2.5.6 Ultimate Scaling of SRAM

The densest possible array structure is $4F^2$, independent of memory technology [13]. This is not possible for SRAM, the reason for this is because the cell is made of multiple transistors which can not be stacked on top of each other. With multiple transistors it is also necessary to spend area to isolate one transistor from another [25]. The array structure varies from $100 \sim 200 \cdot F^2$, depending on the sizing of the transistor widths [25]. According to the ITRS $\sim 140F^2$ structures will be most common, as this represents a good trade-off between area and yield [18, 25].

As discussed in the MOSFET chapter, the ultimate MOSFET is a gate all around structure or a nanowire FET. The ultimate scaling limit for a nanowire FET is given by direct source to drain tunnelling, and as discussed in the MOSFET chapter these current are predicted to dominate at gate lengths shorter than 6 nm, for silicon substrates [31].

Threshold voltage variations is increasing with reduced gate lengths as seen in figure 2.6 on page 18, and this could be a limiting factor for SRAM designs at gate lengths longer than 6 nm. However, it was recently shown that FinFET SRAMs with sufficient read static noise margin is plausible at gate lengths as low as 10 nm [28].

2.6 FeRAM

In this section we will discuss the Ferroelectric RAM (FeRAM), with a focus on embedded applications and ultimate scaling limits. Similar to the presentation of DRAM, we will first briefly discuss the basic operation of the cell to easier identify challenges and demands of the FeRAM cell. The standard FeRAM 1T1C structure has many similarities to the 1T1C DRAM cell. The most important difference is that the FeRAM uses a ferroelectric material in the capacitor which allows for non-volatile operation [13].

2.6.1 Basic Operation

The 1T1C structure of the ferroelectric cell is shown in figure 2.20. As discussed in section 2.6 of my earlier work [1], the structure is quite similar to a 1T1C DRAM cell [13]. The difference is that the regular capacitor used for DRAM, is now a ferroelectric capacitor in FeRAM, and that the plate line of the FeRAM requires pulsing for writing and reading. A ferroelectric material is a material which

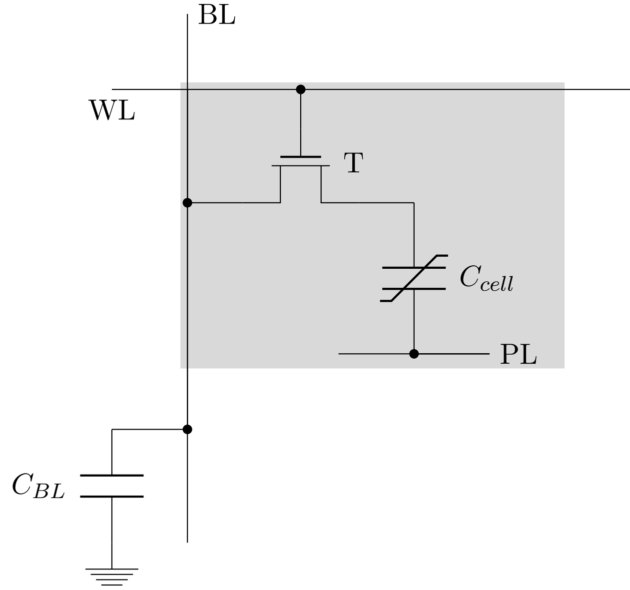


Figure 2.20: Schematic of a FeRAM cell. The cell consists of an access transistor and a ferroelectric capacitor. The cell can be addressed through the bit line and word line.

exhibits a spontaneous polarization $\mathbf{P_S}$, which can be reoriented by an applied external electric field [45]. Figure 2.21 shows a typical ferroelectric hysteresis loop which is exploited in order to create a non-volatile memory [1, 13].

To **write** a cell, a voltage pulse is applied to the word line. During the first half of this pulse the plate line is also addressed, as seen in figure 2.22. To write a zero the bit line voltage is kept at a ground potential. This leads to a to a voltage of $+V_{DD}$ while the plate line voltage is applied. When reducing the plate line voltage, the polarization will be left at a state $+P_r$, indicating a logic zero, as seen in the bottom part of figure 2.22. To write a one, the bit line is addressed during the whole period of the write pulse (see figure 2.22). This leads to a voltage of $-V_{DD}$ at the end of the write pulse (i.e. when the plate line pulse is not applied any more). When the bit line voltage is reduced, a polarization of $-P_r$ will be left on the capacitor indicating a logic one.

To **read** a cell, the bit line is floated, while a positive voltage V_{DD} is applied to the plate line while the word line is active [13] to make the channel conducting, see figure 2.23. The voltage applied to the plate line is divided between the ferroelectric cell capacitor and the parasitic bit line capacitor. The bit line capacitance C_{BL} is usually much larger than the ferroelectric capacitance. This causes most of the applied voltage to be across the ferroelectric capacitor [13]. As shown in

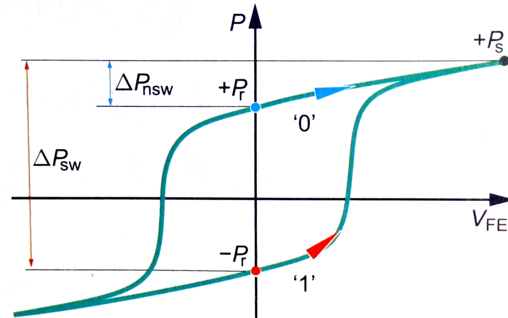


Figure 2.21: Hysteresis P-V Loop of a ferroelectric capacitor. The remnant polarization $+P_r$ and $-P_r$ is shown, which represents the logic states zero and one respectively. ΔP_{nsw} and ΔP_{sw} denote the change in polarization for a non-switching event and a switching event when a voltage $+V_{FE}$ is applied. From [13].

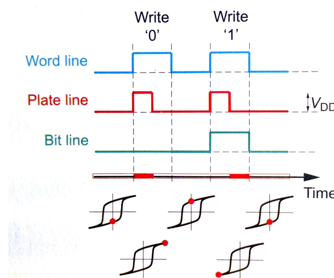


Figure 2.22: Timing diagram for writing a FeRAM cell along with the polarization state for the ferroelectric capacitor. From: [13].

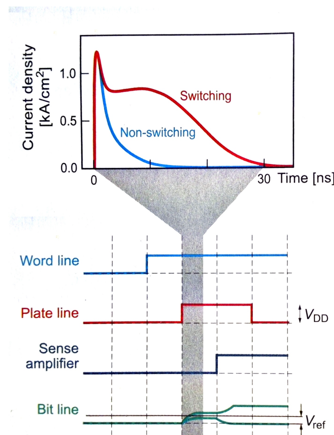


Figure 2.23: Timing diagram for reading a FeRAM cell. The current-time diagram is shown in the top figure. The integrals of the switching (red) and the non-switching (blue) represent ΔP_{nsw} and ΔP_{sw} respectively. Under: the timing diagram of the FeRAM cell is shown. When the plate line is activated a current flows from the cell capacitor to the bit line capacitor. The sense amplifier is activated to translate the voltage level on the bit line back to digital information. The write-back operation is not shown. From: [13].

section 2.6.2 of my earlier work, the bit line voltage after a read operation can be approximated with [1]:

$$V_{BL} \approx \begin{cases} \Delta P_{nsw} \frac{A}{C_{BL}} = V_{BL}^{(0)} & \text{if zero was stored} \\ \Delta P_{sw} \frac{A}{C_{BL}} = V_{BL}^{(1)} & \text{if one was stored} \end{cases} \tag{2.42}$$

where ΔP_{nsw} and ΔP_{sw} is shown in figure 2.21. The $\Delta P_{nsw} = P_S - P_R$, is the non-switching case, and $\Delta P_{sw} = P_S + P_R$ represents the case where the capacitor is switched with a voltage V_{FE}. P_S is the polarization at $V = V_{FE}$, P_R is the polarization at $V = 0$ and A is the area of the ferroelectric capacitor.

After the bit line has settled a sense amplifier is activated in order to drive the bit line to full V_{DD}, if the voltage on the bit line is $V_{BL}^{(1)}$, or 0 V ,if the voltage is $V_{BL}^{(0)}$ [13]. Since the read operation is destructive, each read has to be followed with a successive write-back.

2.6.2 Main Challenges in FeRAM Design

The main challenges in FeRAM design have been to refine the processing steps in order to make sure that it is compatible with the CMOS process [10]. The materials

that has been most studied for use as ferroelectric in capacitors, are lead zirconate-titanate, Pb $(Zr_xTi_{x-1})O_3$ (PZT), and strontium bismuth tantalate, $SrBi_2Ta_2O_9$ (SBT) [13]. To be compatible with the CMOS process, a low crystallization temperature of the ferroelectric material is the most critical parameter. However, a high remnant polarization and a low coercive voltage is important for low power devices. While the ability to withstand more than 1×10^{16} read/write cycles is important to compete against DRAMs in personal computers [13].

Another main challenge for the FeRAM, in order to allow high density storage, is the introduction of 3D ferroelectric capacitors. This is considered a large challenge in FeRAM design, and according to the ITRS, 3D ferroelectric capacitors will first be introduced in 2021 [14].

2.6.3 Embedded FeRAM

There is no clear divergence in embedded and standalone FeRAM technology as there is for DRAM. There reasons for this are that the FeRAM is a younger technology and the fact that 3D capacitors, which require many extra processing steps, is not yet implemented [10]. The main issues for embedded FeRAM design are thus similar to standalone. It is important to achieve a small cell, while keeping a large ferroelectric capacitor. A large capacitor is needed to ensure high reliability; however, the cell must still maintain compatibility with the CMOS process [46].

For embedded FeRAM three different configurations of the 1T1C cell are discussed. In figure 2.24a we see the capacitance under bit line (CUB) structure, it has a good compatibility with the CMOS process, but as we see in the figure it needs a deep bit line-contact to connect the cell and the bit line. This cell configuration gives less room for the ferroelectric capacitor, which will give a small signal and reliability degradation [46]. In figure 2.24b we see the capacitance over bit line (COB) structure. As we see, this eliminates the need for a deep bit line contact. However, in this structure the bit line must be made of tungsten (W) instead of copper (Cu) or aluminium (Al), to endure the thermal annealing of the PZT capacitor crystallization [46]. Because of this the COB structure loses compatibility with the CMOS process.

The third candidate for embedded FeRAM is the capacitance coupled bit line (CCB). As we see in figure 2.24c, the positions of the bit line and plate line are swapped. The top electrode of the ferroelectric capacitor is connected without a bit line contact. This solves the problem for the CUB and CCB structures, and achieves a large capacitor while still using copper/aluminium for the bit line.

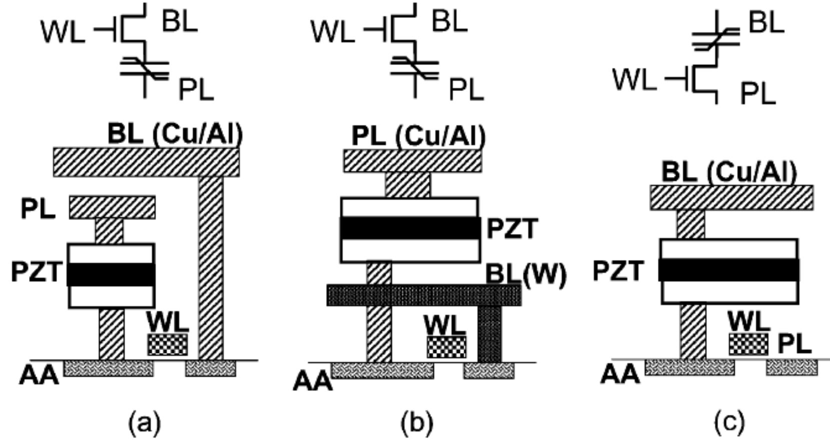


Figure 2.24: Different configurations for the embedded 1T1C FeRAM cell. (a) Capacitance under bit line (CUB) cell. (b) Capacitance over bit line (COB) cell. (c) Capacitance coupled bit line (CCB) cell. AA denotes the Active area of the access transistor. From [46].

Thus the CCB has the best features for embedded FeRAMs, but it has two serious operating problems. One of the problems is an under-shoot of the cell when accessing another cell. However, this can be solved by careful timing of the applied signals. The other problem is direct capacitor coupling, which give rise to a static disturbance bias to unselected cells [46]. Takashima et al. [46] suggested a ladder FeRAM architecture to solve this problem, their solution consists of a reset transistors which is placed in parallel with a block of 1T1C cells to short circuit these cells, and remove the parasitic capacitance coupling.

2.6.4 Ultimate Scaling of FeRAM

In this section we will discuss the parameters of ferroelectric capacitors, how they change when the cell is scaled down, and if there are some factors limiting the scaling.

Although 3D ferroelectric capacitors are difficult to manufacture, long term scaling will eventually lead to $4F^2$ structures for FeRAM, similar to those discussed for DRAM and sketched in figure 2.16 on page 37 [13]. Stable ferroelectric phases has been shown for ferroelectric films down to 2-3 unit cells, corresponding to a ferroelectric thickness of approximately $d = 1\,\text{nm}$ [47, 48], implying that there is no intrinsic ferroelectric size effect.

The coercive field E_C increases when the cell is scaled down, as it follows the

dependence [13]:

$$E_C \propto \frac{1}{d^{\frac{2}{3}}} \tag{2.43}$$

However, this does not impact the FeRAM operation as the coercive voltage V_C is reduced when the thickness is scaled down, because it is given by [13]:

$$V_C = E_C d \propto d^{\frac{1}{3}} \tag{2.44}$$

Assuming that a supply voltage of $V_{DD} = 1.5 \sim 2 \cdot V_C$ is needed for complete switching [13], the supply voltage can also be lowered when the thickness is reduced, allowing for reduced power consumption as the cell is scaled down.

Following the same arguments as for DRAM, with pedestal capacitors as illustrated in figure 2.17 on page 38, the limiting factor will be the sum of the inner electrode and the thickness of the ferroelectric material. From a ferroelectric point of view the thickness, d, of the ferroelectric material could be scaled down to 1 nm, which would result in a minimum feature size of 2 nm. However, at this length scale tunnelling and thermionic emission could affect the cell operation, therefore a more realistic thickness limit would be around $d = 5\,\text{nm}$ [13], similar to the limit for DRAM. Thus the geometry based scaling limit will be similar for DRAM and FeRAM, and will be at a feature size $F = 12\,\text{nm}$ [13].

2.7 MRAM and STT-MRAM

In this section we will look at the similarities and differences in the operation of the conventional field switching Magnetic RAM (MRAM) and the more advanced spin-torque transfer MRAM (STT-MRAM). The STT-MRAM is expected to take completely over for the conventional MRAM by 2016 [10]. Further, we will briefly discuss embedded MRAM and STT-MRAM and finally we will look at the ultimate scaling limits for STT-MRAM.

2.7.1 Magnetic Properties in Layered Structures

As discussed in section 2.7 of my earlier work [1], both MRAM and STT-MRAM uses layered magnetic structures as storage elements. We also discussed that in a magnetic thin film there are two possible stable directions for the magnetization. The reason for this is that some directions are more energetically favourable than others [49]. These directions can either be in the thin film plane, or perpendicular to the thin film. Whether the in plane or perpendicular direction are preferable

depends on thickness of the thin film, the strength of the magnetic field and which materials that make up the interface [1, 49].

In my earlier work we further discussed that in a layered magnetic structure the resistance depends on the relative directions of the two magnetic layers in a magnetic tunnel junction (MTJ) [1]. A magnetic tunnel junction is a layered magnetic structure which exhibits the tunnel magneto-resistance effect (TMR). The TMR effect is an effect that occurs when two ferromagnetic materials are separated with an isolating or semiconducting material [49], as shown in the upper part of figure 2.25. This structure has a difference in the density of states depending on the directions of the magnetic layers [49]. As we see in figure 2.25, this gives a lower resistance in the parallel configuration of the magnetic layers. There is now a good match between initial and final states for the tunnelling process, compared to the anti-parallel configuration [1, 49]. The strength of this change in resistance is often given as the TMR-ratio, which is defined as [49]:

$$\frac{\Delta R}{R_P} = \frac{R_{AP} - R_P}{R_P} \tag{2.45}$$

where R_P is the resistance of the parallel state, and R_{AP} is the resistance of the anti-parallel state. In MRAM and STT-MRAM, the cell consists of a magnetic tunnel junction. Where one of the layers is pinned in a fixed direction (reference layer), while the other layer (storage layer) can be switched to be either parallel or anti-parallel to the reference layer.

2.7.2 Field Switching

In the following sections we will discuss different methods for writing, also called switching of the magnetic layers. In the first generation Magnetic RAMs (MRAMs) two orthogonal magnetic fields where applied to switch the magnetization of the free layer, as seen in figure 2.26. The magnetic fields are generated by sending large currents in the two adjacent conductors. The magnetic fields needed to switch the storage layer are given by [13]:

$$H_x^{\frac{2}{3}} + H_y^{\frac{2}{3}} \geq H_K^{\frac{2}{3}} \tag{2.46}$$

where H_x and H_y are the fields generated by the two conductors seen in figure 2.26, and the H_K is the magnetic anisotropy field of the storage layer. To change the cell to the opposite configuration, the direction of the field is changed by sending the current in the opposite direction. The conductor connected to the cell, as shown in figure 2.26, is often called the bit line, while the other conductor in figure 2.26, which is perpendicular to the bit line, is called the digit line.

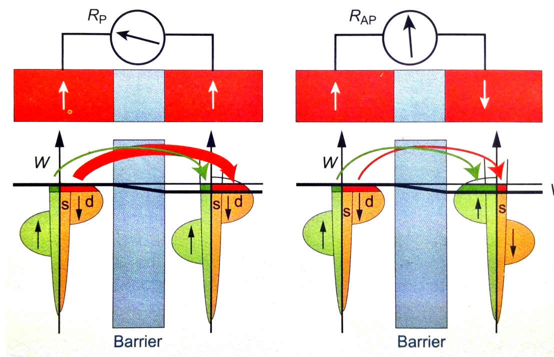


Figure 2.25: If the barrier is much shorter than the spin diffusion length the spin current can be decomposed into spin up (green arrows) and spin down (red arrows). The magnitudes of these two contributions depend on the number of available initial and final states for that spin. In the lower figure, the simplified density of states (DOS) is plotted for 3D metals. As can be seen in this configuration, the best match between the initial and final states around the Fermi level is when the ferromagnetic layers are aligned parallel. This gives the junction a lower resistance for the parallel aligned tunnel junction than for the anti-parallel aligned junction. From: [49].

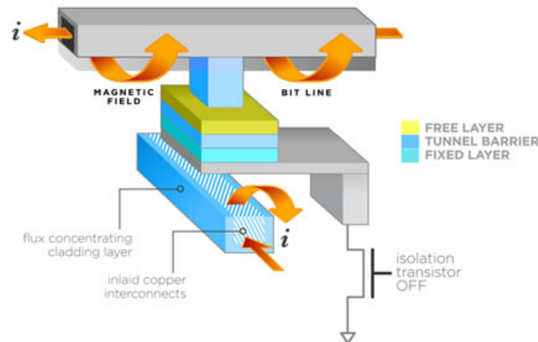


Figure 2.26: Field switching magnetic RAM (MRAM). The magnetic fields are generated from large currents in the conductors, and allows for controlled switching of the free magnetic layer. From [50].

With this configuration it is important that the cells which are only applied H_x or H_y alone not are switched, as this will write cells that are not supposed to be written. To avoid this problem it is important that the H_K field is the same in all the cells in the array [13]. Any defect or dispersion in the shape of the MTJ will result in a broadening in the switching field distribution. This problem is known as the "half select instability" and it sets high demands for the fabrication process [13].

2.7.3 Toggle Field Switching

To circumvent the problem with the "half select instability" a method called toggle switching was introduced by Savtchenko from Motorola [51]. In a toggle MRAM, the magnetic layers are elongated ellipsoids with the long axis pointing 45° with respect to the field from the two conductors, as shown in the lower inset of figure 2.27. This configuration is more stable because the activation energy is increased if only one of the fields is applied [52].

Looking at figure 2.27 we see that the switching is done by rotating the magnetization in steps by [53]:

- First a magnetic field is applied by sending a current through the bit line field. Marked as 1 in figure 2.27
- Then a magnetic field is applied by sending a current through the digit line, marked as 2 in figure 2.27. This field is perpendicular to the bit line field.
- The field from the bit line is then removed. Marked as 3 in figure 2.27.
- When the digit line field then is removed the magnetization will relax into the opposite direction of the initial state.

Both the solid and the dashed arrow around in figure 2.27 switches to the opposite direction. This means that the final state is independent of the initial alignment of the magnetization. This is again why this method is called toggle MRAM.

Because the cell is switched independent on the initial state, the state of the cell has to be sensed before it is written to check if a toggle is necessary. The complication of needing to first sense the state is compensated for by the fact this configuration only need current to be sent in one direction. This makes it possible to optimize transistors for current sinking or sourcing [53].

Although the toggle MRAM solves the problem with the "half select instability", the switching of MRAMs with magnetic fields are poorly scalable [13]. The reason for this is that the energy barrier between the two states must remain high when

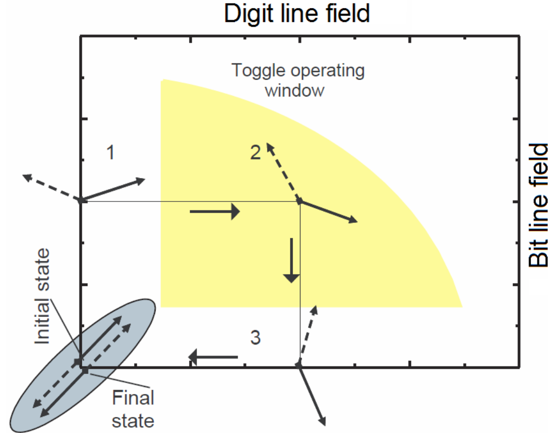


Figure 2.27: Toggle switching of the MRAM cell. To rotate the magnetization the fields are applied and removed following the numbers 1-3. We will follow the square clockwise and in the final state end with a magnetization which is reversed compared to the initial state. From [53].

the cell is scaled down, in order to prevent random thermal switching [13]. Thus the magnetic field required to overcome this energy barrier and switch the magnetization of layers must be correspondingly high. When scaling down the cross-section of the bit line, the current density required to generate the magnetic field increases drastically. Even with some techniques like a cladding around the conductor to divert the magnetic field on the backside of the conductor [53], the current density can reach values as high as the electro-migration limit, $\sim 1 \times 10^7\,\mathrm{A\,cm^{-2}}$ [13]. This is why the STT effect gathered a lot of attention when it was predicted in 1996 [13], as it provides another means of switching the magnetic layer.

2.7.4 Spin-Torque Transfer (STT) Switching

Spin-Torque Transfer (STT) is an effect that makes it possible to switch the free magnetic layer without applying an external magnetic field. The spin-torque transfer effect is the reciprocal interaction of the TMR [49]. A spin polarized current can switch the direction of a ferromagnetic layer. The transverse component of the spin of the electrons is absorbed at the interface between a magnetic and non-magnetic layer, exerting a torque on the magnetic layer. The direction of the current through the MTJ will decide whether a zero or one is written. A comprehensive quantum mechanical treatment of the effects involved is given in [54]. The possibility to switch the cell by applying a current also makes a substantially simpler cell design, as shown in figure 2.28.

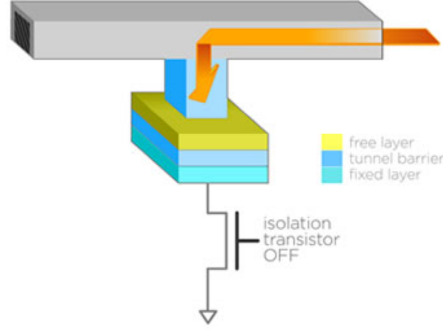


Figure 2.28: Schematic of STT-MRAM cell. The cell is switched by sending a critical current density through the MTJ stack. The difference between writing a one and a zero is the direction of the current. From [55].

It can be shown that the critical current density for which the magnetization switches, due to spin-torque transfer effects, is given by a critical current density. For in plane magnetization it can be expressed as [13]:

$$J_{\text{write,in-plane}} = \left(\frac{2e}{\hbar}\right)\frac{\alpha t_F}{P}\left(\frac{\mu_0 M_s^2}{2} + 2K\right) \tag{2.47}$$

where P is the current polarization (typically about 50 % for most materials), K is the uniaxial anisotropy of the storage layer, M_s its spontaneous magnetization, α its Gilbert damping and t_F its thickness. μ_0 is the vacuum permeability, $\hbar$ is the reduced Planck constant and e is the electron charge. The most important aspect of this result is that the switching is now given by a critical current density instead of a critical current. This implies that the total current needed to switch the cell is reduced as the cell is scaled down [13]. The lowest achieved values for the current density for in plane magnetization are in the order of $\sim 2 \times 10^6\,\mathrm{A\,cm^{-2}}$ [56].

2.7.5 Reading the Cell

Figure 2.29 shows an equivalent circuit of a 1T1R MRAM or STT-MRAM memory cell when reading. To read the cell, the resistance of the MTJ should be sensed. This can be done by applying a voltage on the word line in order to make the channel conducting, and then apply a small voltage V_{read} on the bit line. The resulting current through the bit line will then be [13, 38]:

$$I_{BL} = \begin{cases} \frac{V_{\text{read}}}{R_{BL}+R_{FET}+R_{AP}} = I_{BL}^{(0)} & \text{if zero was stored} \\ \frac{V_{\text{read}}}{R_{BL}+R_{FET}+R_{P}} = I_{BL}^{(1)} & \text{if one was stored} \end{cases} \tag{2.48}$$

where R_{BL} is the parasitic resistance of the bit line and R_{FET} is the resistance of the transistor channel. This current can be sensed with a sense amplifier and converted to a digital signal. For STT-MRAM it is important that this read signal is not too large, or else it might switch the cell.

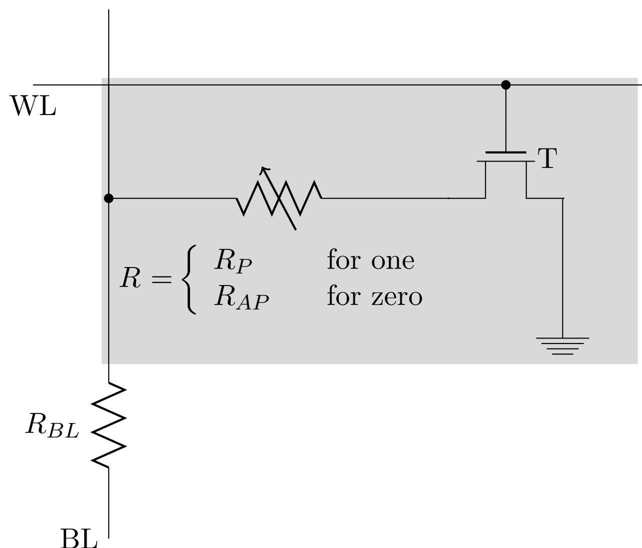


Figure 2.29: Equivalent circuit when reading a MRAM or STT-MRAM cell, the cell can be accessed through the bit line and word line. The cell resistance, R, can have two possible values depending on whether the ferromagnetic layers are stored in a parallel or anti-parallel alignment.

2.7.6 Embedded MRAM and STT-MRAM

Embedded field switching MRAM has been shown to be possible with a front end CMOS process and a back end magnetic process [13]. As shown in figure 2.30, the MTJs can be embedded in the top metal layers and connected to the access transistors by vias [57]. This technology was developed by Everspin and is especially designed to be integrated with microcontrollers [57]. The reason for placing the magnetic tunnel junctions in the top metal layers is to make sure that the integration of extra metal layers, above the memory array, does not affect the MTJ [20].

The STT-MRAM is a newer technology, so no such specialized embedded product is yet the available, but it is expected to be possible [20], and some prototypes exists [58]. The integration of magnetic tunnel junctions is easier to integrate than for example special capacitors for embedded DRAM. The process temperatures for

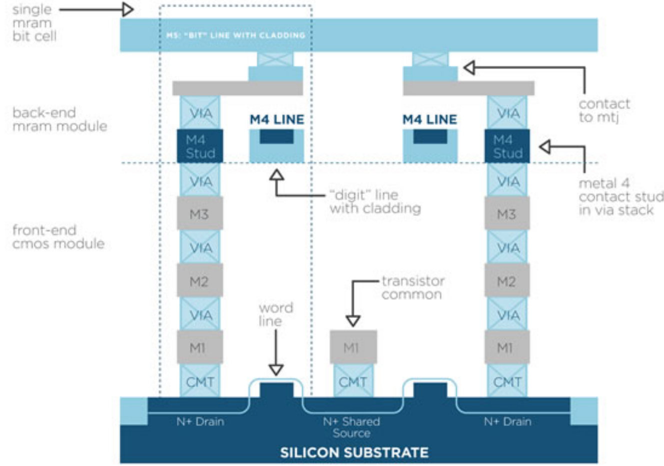


Figure 2.30: Cross section of a MRAM cell integrated in the top metal layers of a device. It is possible to embed the MRAM technolgy in a two step process: First a frond-end CMOS module and then a back-end magnetic module with just a few added masks. From [57].

MTJs are below $\sim 350\,°\mathrm{C}$, causing no conflict with the CMOS process [20]. The main challenge for embedded STT-MRAM is to ensure good operating margins at process, temperature and voltage corners in order to maintain a high yield [20].

2.7.7 Ultimate Scaling of STT-MRAM

In this section we will discuss limits for the ultimate scaling of STT-MRAM. As mentioned earlier, the field switching MRAM is considered to not be scalable beyond the 65 nm node due to too high switching currents [10, 13].

As discussed in section 2.7.7 of my earlier work [1] there is a chance that the thermal fluctuations can flip the magnetization layer. To suppress these fluctuations and maintain a retention time of 10 years, it was shown that a energy barrier, ΔE, larger than $50k_BT$ is needed. Where k_B is the Boltzmann constant and T is the temperature. This gives the relation [1]:

$$KV = \Delta E > 50k_BT \tag{2.49}$$

where V is the magnetic volume of the storage layer and K is magnetic anisotropy per unit volume. $50k_BT$ is the limit for a single cell in a multi-megabit array, it can be shown that this limit increases to $\sim 70k_BT$ [13, 59]. As the dimensions cell

is scaled down this criteria gets harder to fulfil, and a higher magnetic anisotropy K is needed. However, as we see in equation (2.47) this will also increase the current density needed to switch the MTJ. This increase in current density means that larger access transistors are needed and can ultimately lead to device failure, because the MTJ can undergo breakdown if submitted to bias voltages larger than $\sim 1.2\,\mathrm{V}$ [13].

To reduce the problem with increasing current density, a suggested solution is to use MTJ stacks with perpendicular to plane magnetization [13]. The current density is reduced for perpendicular to plane magnetization because the two terms in equation (2.47) partially cancels. With perpendicular to plane magnetization the critical current density needed to switch the ferromagnetic layers are given by [13]:

$$J_{\text{write,perpendicular-to-plane}} = \left(\frac{2e}{\hbar}\right)\frac{\alpha t_F}{P}(2K_{eff}) \tag{2.50}$$

where K_{eff} is the effective perpendicular anisotropy, which takes into account the perpendicular anisotropy of the bulk or interfacial origin minus the demagnetizing energy of the layer. Much research is currently put into finding suitable materials for a perpendicular to plane MTJ [10,13], and ITRS expects perpendicular to plane magnetization to be introduced in STT-MRAM production from 2016 [14].

The scaling limit for the STT-MRAM therefore looks to be the access device, in order to get enough current to switch the cell [13, 60]. Using $I = J \cdot A$ and $V = A \cdot t_F$, where A is the area of the MTJ, we can insert equation (2.49) into equation (2.50) to get the total current through the magnetic tunnel junction, as a function of the energy barrier ΔE [13]:

$$I_{\text{write,perpendicular-to-plane}} = \left(\frac{4e}{\hbar}\right)\frac{\alpha \Delta E}{P} \tag{2.51}$$

Assuming $\Delta E \sim 70k_BT$ [61], $T = 300\,\mathrm{K}$, $\alpha \sim 0.01$ and $P \sim 0.8$ [60] yields a minimum current in the order of $25\,\mu\mathrm{A}$ for a simple MTJ magnetized perpendicular to the plane. It is possible to double the STT efficiency by using a double barrier scheme, so that the minimum current drops to about $12\,\mu\mathrm{A}$ [13]. This is not far away from the lowest switching current achieved of $29\,\mu\mathrm{A}$, achieved by Gajek et al. [62] for a perpendicular to plane STT-MRAM.

If we further assume that the access transistor can deliver a current per device width of $\sim 1000\,\mu\mathrm{A}\,\mu\mathrm{m}^{-1}$ [60], we get the lowest possible channel width for the access transistor of about $12\,\mathrm{nm}$ [13]. As mentioned for the other technologies, the densest array for a given feature size is $4F^2$, this is also possible for STT-MRAM, but will require vertical channel access transistors (similar to DRAM) to deliver enough current [59, 63].

Further down scaling could be possible by solving the retention problem by a thermally assisted MRAM (TA-MRAM). In this approach an antiferromagnetic layer with a blocking temperature in the order of 200 °C, pins the magnetic layer. The blocking temperature is the temperature below which the anisotropy is sufficiently large to block thermal motion of the magnetic moments with respect to the crystal axes [49]. This antiferromagnet will give reduced activation energy when the MTJ is above this blocking temperature, and gives excellent thermal stability at room temperature [13]. To write the cell it first needs to be heated above the blocking temperature. This can be achieved by applying a large current to induce Joule heating. Joule heating is the process of heating a material by sending a current through it. The process concerning thermally assisted MRAM needs further refinement to be viable for mass production, but would offer the ultimate scalability as in magnetic recording technology [13, 60].

2.8 PCRAM

In this section we will look at Phase Change RAM (PCRAM) with a focus on different aspects of embedded PCRAM and looking at ultimate scaling of the PCRAM cell. However, first we will briefly repeat some of the basics about PCRAM and the physical principles involved. This was also done in section 2.8 of my earlier work [1], but we will have a short review of the most important results here, for completeness and to get a better understanding of the limiting factors, when discussing the more advanced material.

PCRAM utilizes a difference in resistance between an amorphous and crystalline state to store information. Almost all solid materials can be prepared in an amorphous or crystalline state. What is special about phase change materials is a large change in resistance between the two phases, and that the crystallization of the amorphous state is particularly fast [13]. For RAM applications the crystallization time is below 100 ns [9], and values as low as 1 ns has been observed [64]. This combination of abilities is restricted to a set of semiconductor alloys, mostly comprised of chalcogenides (selenides and tellurides) and several antimony compounds such as Ge-Sb and Ga-Sb based materials [13]. The most common used material for RAM applications is Germanium-Antimony-Tellurium, $Ge_2Sb_2Te_5$ (GST) [65]. Phase change materials are not only used in emerging RAM technology, but also widely used in rewritable CD, DVD and blu-ray discs. But in these applications, the change in resistance is not important; it is a change in the refractive index which is utilized [13].

2.8.1 Properties of Phase Change Materials

As discussed in my earlier work, the properties of the phase change material have a well-defined atomic origin [1, 13]. It is because of a special type of resonant bonding [13], which needs a sufficiently ordered atomic arrangement. This order is only present in the crystalline phase, and this leads to large deviations in properties between the crystalline and the amorphous state [1].

When a potential is applied to the amorphous phase, the conductivity can be described by electronic hopping transport of carriers by thermal emission over a potential barrier [13]. This leads to a reduction in resistance with increased temperature. In addition, an applied electric field will reduce the energy barriers. At a material dependent threshold field, the electrons gain so much energy that the local field collapses, leading to a negative differential resistance. The process is reversible if the electric field is turned off fast enough, so that negligible Joule heating occurs, this leads to a phase transition [1].

2.8.2 Basic Operation

A typical 1T1R cell is shown in figure 2.31, it consists of one resistor made of a phase change material, and one access transistor. If the cell is in a crystalline state, the cell is referred to as set, corresponding to a logic one. Contrary, if the cell is in the amorphous state, it is referred to as reset, corresponding to a logic zero. The phase change material is often structured as shown in figure 2.32a, a narrow bottom electrode called the heater is placed under the phase change material in order to limit the path where the current can flow, thus confining the heat. This again leads to the mushroom shaped area of the programmable region above this heater, which has given the cell the name mushroom cell [13].

To reset the cell, a high temperature (above 600°C for many materials [13]) over a short period is needed, see figure 2.32b, in order to suppress Joule heating. This is done with a corresponding high and short current pulse. In order to set the cell it is heated above the crystallization temperature for a longer time so that recrystallization can happen, as seen in figure 2.32b. A crystallization temperature for a good phase change material is around $150 \sim 200\,°\mathrm{C}$ [13].

Similar to the MRAM technologies, the cell can be read by applying a small voltage V_{read} and sense the current through the bit line. The read voltage must be low enough so that no change in the state of the cell happens. And as equation (2.48) which gives the bit line current for MRAM, the current through the bit line for

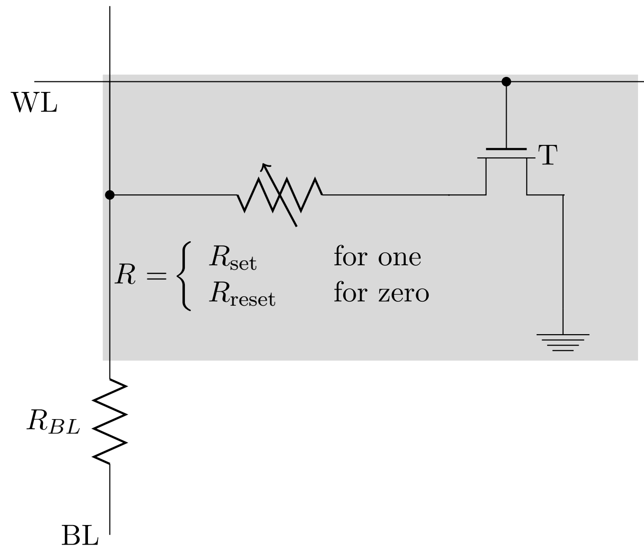


Figure 2.31: Schematic of a 1T1R PCRAM cell. The cell can be accessed through the bit line and the word line. The resistance is made of a phase change alloy that can either be in a high resistance amorphous state denoted R_{reset}, or a low resistance crystalline phase denoted R_{set}.

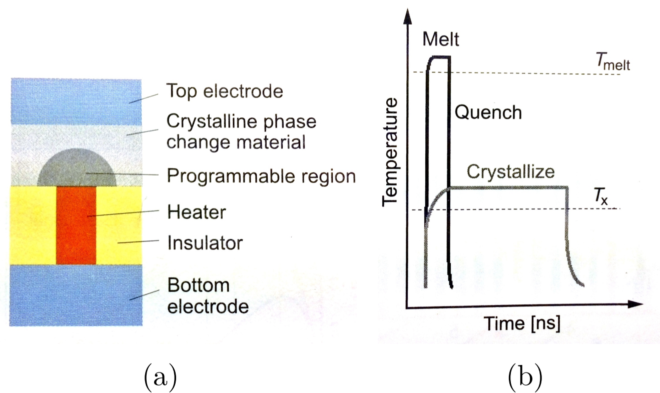


Figure 2.32: (a) Schematic of a typical mushroom PCRAM cell, the name comes from the shape of the programmable region above the heater. The heater is also called bottom electrode contact (BEC). (b) Temperature vs. time for a PCRAM cell during set and reset. From [13].

the PCRAM is [13, 38]:

$$I_{BL} = \begin{cases} \frac{V_{\text{read}}}{R_{BL}+R_{FET}+R_{\text{reset}}} = I_{BL}^{(0)} & \text{if zero was stored} \\ \frac{V_{\text{read}}}{R_{BL}+R_{FET}+R_{\text{set}}} = I_{BL}^{(1)} & \text{if one was stored} \end{cases} \tag{2.52}$$

where R_{BL} is the parasitic bit line resistance and R_{FET} is the equivalent resistance of the FET channel. R_{reset} and R_{set} is the resistance of the amorphous and crystalline phases respectively.

2.8.3 Multi-Level Storage

To maximize the bit density, it is very beneficial to be able to store multiple bits in one cell, this is called multi-level cell (MLC). Storing one bit in a cell is in contrast called single-level cell (SLC). Storing multiple bits in one cell has been done with large success for flash memories [10].

There are two methods for achieving multi-level storage for PCRAM: One way is to use multi-bit, where each memory element is programmed into intermediate resistance values in order to store multiple bits. The other possibility is to make multiple layers of phase change material which changes phase at different temperatures, but share the same addressing and sense amplifier circuitry [66]. However, due to the additional costs of multi-layers concerning extra masks needed, reduced yield and reliability concerns, multi-layer does not pay-off as much as multi-bit systems, as we see in figure 2.33 [66]. Therefore, multi-bit systems are the technology which is anticipated to be used for multi level cells [10].

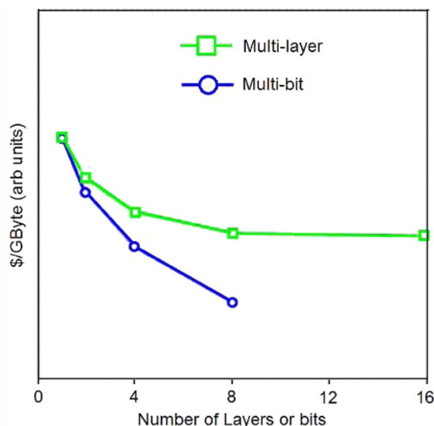


Figure 2.33: Cost predictions for different multi-level storage techniques. From [66].

Multi-bit storage is achieved by either partially setting or partially resetting a cell, this leads to intermediate values of the resistance [13, 67]. A partial reset is shown in figure 2.34a. Here, it is a change in the volume of amorphous part of the phase change material which gives the partial resistance. A partial set is shown in figure 2.34b. Here, crystalline filaments in the amorphous material are created, which acts as low resistance channels where the current can flow [67].

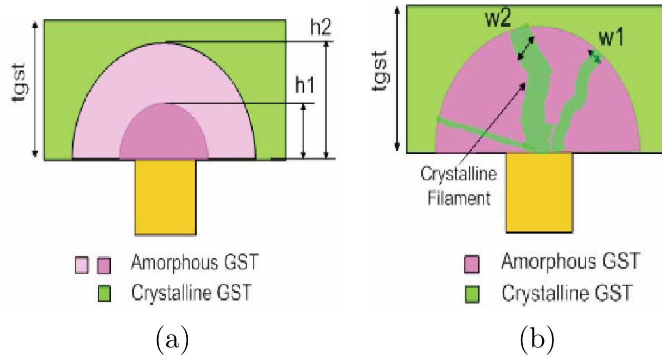
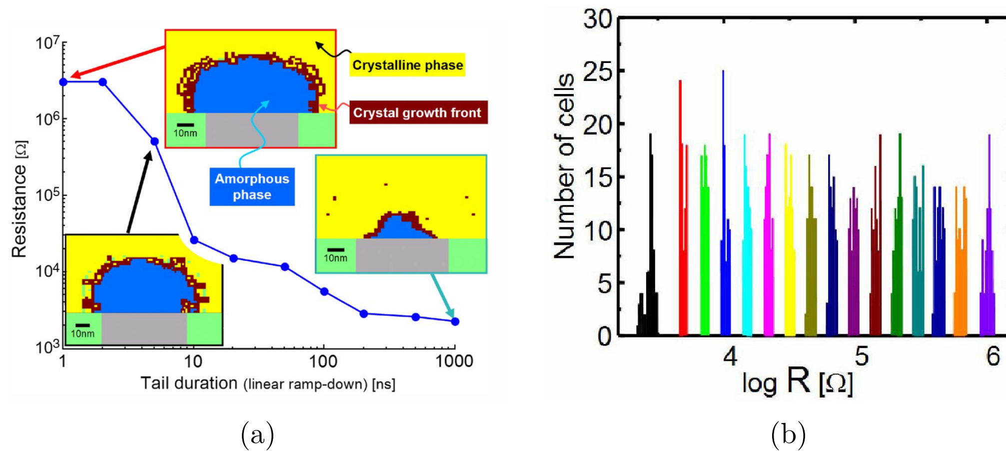


Figure 2.34: Different ways of achieving multi-bit cells. (a) Partial reset. Depending on the current which were used to reset the material, different volumes of amorphous phase change material is achieved. A lower volume of amorphous material corresponds to a lower resistance. Therefore, the case where the amorphous volume has a higher height, h_2, will have a higher resistance than if the height is h_1. (b) Partial set. Different resistances are achieved by adding more and wider low resistance filaments where the current can flow. The filament with a larger width, w_2, has a lower resistance than the filament with a smaller width w_1. From [67].

Partially resetting has been achieved by varying the amplitude and the trailing edge of the current pulse [68], this leads to different sizes and shapes of the material, which is in the amorphous phase and thus an intermediate resistance, as shown in figure 2.35a. With an iterative writing algorithm, which is called write and verify, up to sixteen resistance level has been demonstrated, as shown in figure 2.35b [68]. This allows for storage of four bits per cell. Similar results have been achieved with the partial set method [67, 69, 70]. The iterative write and verify algorithm used by Nirschl et al. [68] is showed in figure 2.36. This algorithm is similar to those used in NAND flash technology, and it is needed for both partial set and partial reset methods in order to achieve tight and separable resistance distributions. The resistance distributions are relative wide if not written with a write and verify algorithm, because of nanoscale variations in the cell structure and the randomness associated with the thermal writing process [66].



(a) (b)

Figure 2.35: (a) Electro-thermal simulations of the phase change material shows that by adjusting the length of the ramp down of the reset current pulse, it is possible to achieve different shapes and sizes of the part of the material which is in the amorphous state. As shown, this leads to intermediate resistance values. (b) By using an iterative algorithm to determine the right amplitude and duration of the current pulse, tight resistance distributions where achieved for a 10x10 array. This allows for 4 bits per cell. From [68].

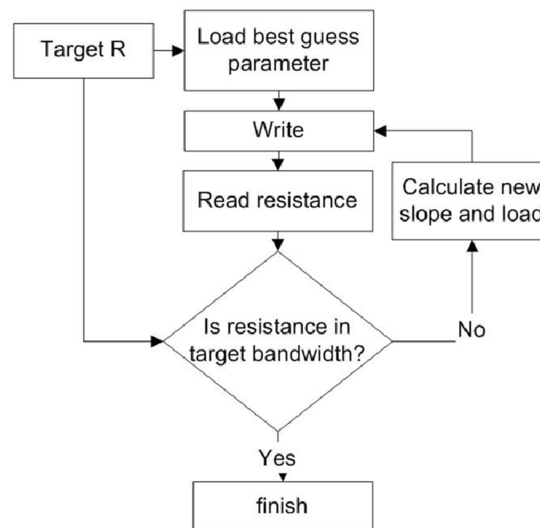


Figure 2.36: Write and verify algorithm. An algorithm like this is needed in order to achieve tight and separable resistance distributions. From [68].

However, a drift in the resistance levels of the amorphous phase change material complicates the implementation of multi-level storage [13]. This drift is not of significance when the phase change is used for only storing one bit, because the crystalline phase is very stable [13], but it can be a problem for multi-level storage.

A problem with a write and verify algorithm which is used for multi-level storage is an increase in the writing time, as the cell is written and read multiple times. This is worsened by the fact that there is no limit for how long a write operation can take. The reason for this is that there is always a chance that a write operation fails, which again can lead to many extra iterations. If many extra write operations are needed it can drastically reduce the overall performance of the system [69, 71]. A suggestion to work around the latter problem is to allow for the cells which require more than a set number of iterations to be left in a wrong resistance state. Then instead let error correcting circuitry (ECC) correct the error when the cell is read [69].

2.8.4 Embedded PCRAM

The PCRAM has proven compatibility with the CMOS process [13, 21] and embedded PCRAM has been proposed integrated in the top metal layers as a back end process [21] (similar to that discussed for MRAM) and at the substrate level in between the back-end and front-end of the CMOS process [72].

When the PCRAM is embedded in the top metal layers, as shown in figure 2.37, it can be done with no adjustments to the CMOS process [21]. The CMOS back-end process uses several temperatures over 400 °C, these temperatures can deteriorate the GST material properties [21]. However, by placing the phase change material in the top metal layers, this problem is avoided, and it can be done by using only two extra masks [21]. A drawback with this approach is that the long vias needed will add to the parasitic capacitances and resistances.

When the PCRAM is embedded between the front-end and the back-end CMOS process, as shown in figure 2.38, small adjustments in the back-end flow is necessary in order to reduce the thermal budget, in order to guarantee GST integrity. Annunziata et al. [72] showed that this was possible with only three extra masks. They also showed that six metal levels above the GST was possible with no impact on device performance [72].

A problem for PCRAM in embedded applications is the need for the ability to operate at temperatures that are substantially above room temperature. Typical demands are up to 85 °C for consumer applications, or even 150 °C for automotive

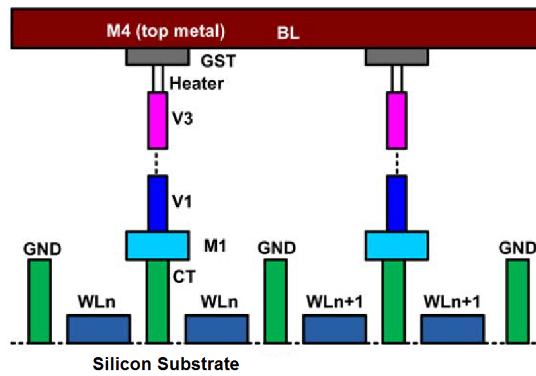


Figure 2.37: Cross section of a PCRAM embedded in the top metal layer of a device. The PCRAM is embedded in the fourth metal level, which in this example is the top metal layer. From [21].

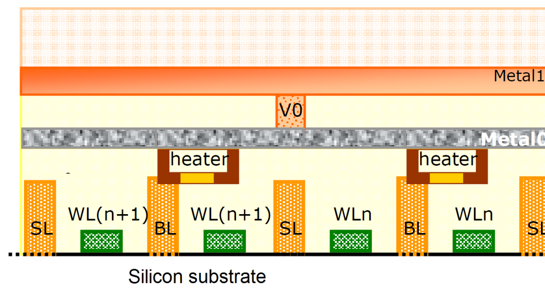


Figure 2.38: Cross section of a PCRAM embedded in between the front-end and back-end CMOS process. From [72].

applications [13]. More than 10 year retention at 85 °C has been showed in multiple publications [21, 72, 73], but increasing this has been difficult. ITRS predicts that operation at temperatures up to 125 °C could be possible from 2016 [74].

2.8.5 Ultimate Scaling of PCRAM

The phase change materials show remarkable scaling properties. Even today the density in most applications is not limited by the phase change material itself, but by the access device [10]. The reason for this is the relative high reset current needed, most scaling strategies are therefore focused on reducing this current [13]. Fortunately the reset current can be reduced by reducing the volume of the phase change material that needs to be heated, and therefore the current needed to switch is reduced by scaling down the cell dimensions.

The volume of the storage element which are switched can also be further reduced by scaling down the contact area between the bottom electrode (heater) and the phase change material [13]. This was taken to the extreme by Liang et al. [75], which used carbon nanotubes (CNT) as heater electrodes. They achieved a reset current of 1.4 µA with a carbon nanotube with diameter of 1.2 nm as a contact. As seen in figure 2.39, the small contact area between the carbon nanotubes and the phase change material allowed for a large reduction in the reset current [76].

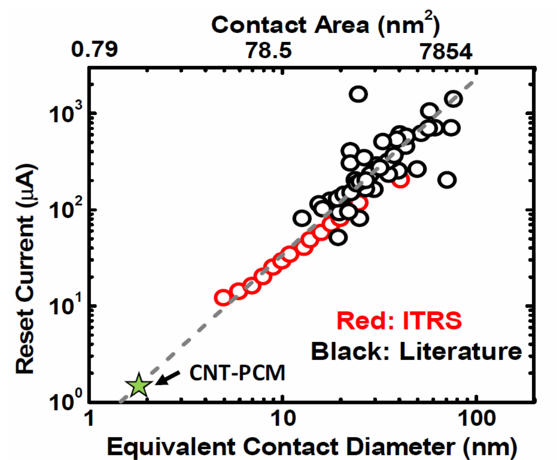


Figure 2.39: Reset current vs equivalent contact diameter for PCRAM devices. The data is collected from the ITRS (red) and from literature (black). The star marks the work of Liang et al. [75] which used carbon nanotubes (CNT) as heater electrode. From [76].

Scaling studies on the phase change material shows that size effects of the phase

change material is relevant when the material is scaled to thicknesses lower than 10 nm [13, 66]. Below this thickness the material properties like the crystallization temperature depend strongly on the interface, because the nucleation starts at the interface [13, 66]. This represents both challenges and opportunities for the phase change material. The reason for this is that with interface effects, the device becomes more complex. But it also allows for careful engineering of the interface properties in order to achieve a desired set of characteristics [66]. Phase change nanoparticles were reported down to sizes of 2 nm, which were stable at room temperature in the amorphous and crystalline state [13]. This looks to be the ultimate scaling limit, as at least a few atoms are needed in order to form a crystalline phase. 2 nm corresponds to about three times the lattice constant [13].

As discussed for the other memory technologies, $4F^2$ is the densest possible array structure for a given feature size. This has been achieved for commercial PCRAM devices, but it requires a bipolar junction transistor (BJT) as an access device [14]. The reason why BJT transistors are needed is that they can deliver a higher current than a MOSFET with similar dimensions, and as mentioned a relatively high current is needed for the reset operation [14].

3 Methodology – Theoretical Estimates

This section contains the methodology, which are used to do the theoretical estimates of the different aspects of a memory technology, such as area and power consumption. The estimates are done for 2012 and 2017 to be consistent with my earlier work [1]. 2012 represents what is available today, while 2017 represents about five years ahead in time, which is the order of the time period required for a semiconductor company to introduce a completely new technology [2]. The equations used in our estimates are listed along with any assumptions made. This work is a continuation of the model that were developed during my project thesis [1].

All parameters of each equation are listed and tabulated along with its source. The ITRS roadmap is used extensively as a source of information and parameters throughout this work. However, other sources are used when the needed parameters is not tabulated in the roadmap. Since my project thesis all parameters have been revised. The parameters have also been updated to the 2012 version of the ITRS roadmap, which were not yet published when the project thesis was written.

Some partial results are also listed in this section, but for the full results please refer to section 5. The emerging memories that are assessed are: FeRAM, MRAM, STT-MRAM and PCRAM, they are all compared against DRAM and SRAM. All estimates are done for standalone memories. The reason for this is that standalone memories are the driving factor for the memory technology, the embedded memory technologies are expected to follow the same trend [10], with some time lag and some variations as discussed in the theory section.

3.1 Area Consumption

The area consumption is one of the most important parameters for all types of memory [10]. In this section, the area consumption in terms of number of bits per area is calculated. The cell area also impacts the energy consumption, because larger cells needs longer interconnects, as we shall see in the following sections.

The area of one cell is calculated as [14]:

$$\text{Cell area} = X_{\text{AF}} \cdot F^2 \tag{3.1}$$

where F is the technology node or half metal pitch according to the year of interest. X_{AF} is the area factor; it is a factor that tells how dense the cell possibly can be

made. The number of bits per area can be calculated as [14]:

$$\text{bits/area} = \frac{\text{bits/cell}}{\text{Cell area}} \tag{3.2}$$

The estimates for area consumption are shown in tables 3.1 and 3.2.

	Technology node F [nm]	Area factor X_{AF}	Bits per cell	Cell area [nm^2]	Bits per µm^2
DRAM	31	6	1	5700	175
SRAM	32	140	1	143360	7.0
FeRAM	180	23	1	712800	1.4
MRAM	90	51	1	413100	2.4
STT-MRAM	65	20	1	84500	11.8
PCRAM	38	12	1	17328	57.7

Table 3.1: Number of bits per µm^2 in 2012. Adapted from: [14]

	Technology node F [nm]	Area factor X_{AF}	Bits per cell	Cell area [nm^2]	Bits per µm^2
DRAM	18	4	1	1300	769
SRAM	16.9	140	1	39985	25
FeRAM	90	22	1	113400	8.82
MRAM	65	52	1	219700	4.6
STT-MRAM	32	10	1	10240	98
PCRAM	18	6	4	1944	2058

Table 3.2: Number of bits per µm^2 in 2017. Adapted from: [14]

3.2 Estimating the Energy Consumption

In this section the energy consumption of the different memory technologies are estimated. Since the goal is to estimate the energy consumption of a given technology, the emphasis has been made on calculating the energy costs to write/read per bit. This makes it possible to estimate the energy consumption of the technology without being bound to one architecture. Therefore, the peripheral circuitry is not taken into account. However, this sets some limitations on our model. These limitations will be discussed later in sections 5 and 6.

This section has been widely expanded in this work, compared to the project thesis that was written earlier [1]. In the project thesis, the write and read power was estimated after the following expressions [1]:

$$P_{\text{write}} = \left(\frac{1}{2}C_{\text{BL}}V_{\text{BL,write}}^2 + \frac{1}{2}C_{\text{WL}}V_{\text{WL,write}}^2 + E_{\text{cell-write/bit}}\right) \cdot f \tag{3.3}$$

$$P_{\text{read}} = \left(\frac{1}{2}C_{\text{BL}}V_{\text{BL,read}}^2 + \frac{1}{2}C_{\text{WL}}V_{\text{WL,read}}^2 + E_{\text{cell-read/bit}}\right) \cdot f \tag{3.4}$$

where C_{BL} and C_{WL} is the bit line and word line capacitance respectively. $V_{\text{BL,write}}$ and $V_{\text{WL,write}}$ is the voltage swing over the bit line and word line when writing. Similar $V_{\text{BL,read}}$ and $V_{\text{WL,read}}$, is the voltage swing over the bit line and word line when reading. $E_{\text{cell-write/bit}}$ and $E_{\text{cell-read/bit}}$ is the bitwise energy cost to write or read one bit respectively. f is the operating frequency. The main problem with these estimates is that it did not take into account the fact that one generally reads or writes multiple bits at a time, as well not taking into account the energy required to switch the access transistors. As we will see, the models in this work is based on the same idea, but the ideas are refined and further developed in order to get as good estimates as possible.

Thus the equations for estimating the write effect, P_{write}, and read effect, P_{read}, estimations has been expanded and now take into account:

- Charging the capacitance of the word line and bit line.
- The energy it costs to write or read a bit.
- The energy required to charge the oxide capacitance of the access transistors.

How this is done will be presented in the following sections.

3.2.1 Power Consumed by the Interconnects

One larger contributor to the power consumption in a RAM is the parasitic capacitance of the bit lines and word lines. The power consumed by the interconnects is given by [77]:

$$P = \frac{1}{2}CV^2 \cdot f \tag{3.5}$$

where C is the total capacitance, V is the voltage and f is the frequency. So that the power cost for charging the capacitance of the word line while writing or reading is:

$$P_{\text{WL,write}} = \frac{1}{2}C_{\text{WL}}V_{\text{WL,write}}^2 \cdot f \tag{3.6}$$

$$P_{\text{WL,read}} = \frac{1}{2} C_{\text{WL}} V_{\text{WL,read}}^2 \cdot f \tag{3.7}$$

where C_{WL} is the word line capacitance and $V_{\text{WL,write}}$ and $V_{\text{WL,read}}$ is the word line voltage while writing and reading respectively. For a typical RAM you read or write multiple bits at a time. This group of bits is called a word, and the number of bits is called the word size, N_{ws} [13]. Taking this into account, for each time we apply a voltage to a word line we write N_{ws} bits. This means that we have to apply a voltage to the fraction $\frac{N_{\text{ws}}}{\text{bits/cell}}$ bit lines. This gives following equations for estimating the power consumed for charging the bit lines while writing or reading:

$$P_{\text{BL,write}} = \frac{N_{\text{ws}}}{\text{bits/cell}} \cdot \frac{1}{2} C_{\text{BL}} V_{\text{BL,write}}^2 \cdot f \tag{3.8}$$

$$P_{\text{BL,read}} = \frac{N_{\text{ws}}}{\text{bits/cell}} \cdot \frac{1}{2} C_{\text{BL}} V_{\text{BL,read}}^2 \cdot f \tag{3.9}$$

where C_{BL} is the bit line capacitance and $V_{\text{BL,write}}$ and $V_{\text{BL,read}}$ is the bit line voltage while writing and reading respectively.

3.2.2 Power Consumed to Write or Read One Cell

Generally, some energy is required in each cell to write or read it, and it will depend on the memory technology. When we take into account that we write N_{ws} bits at a time the contribution to the total power consumption will be:

$$P_{\text{cell,write}} = N_{\text{ws}} E_{\text{cell-write/bit}} \cdot f \tag{3.10}$$

$$P_{\text{cell,read}} = N_{\text{ws}} E_{\text{cell-read/bit}} \cdot f \tag{3.11}$$

where $E_{\text{cell-write/bit}}$ and $E_{\text{cell-read/bit}}$ is the power required to write or read a cell per bit.

3.2.3 Power Consumed to Open the Access Transistors

We must also take into account the energy required to open the access transistors. When the read or write voltage is applied to the word line, all the transistors connected to that line will be switched. The number of transistors on one word line is equal to the number of bit lines N_{BL}. Similar to the ITRS, we model the energy required to switch the transistors as the energy required to switch the capacitance of the gate oxide [14]. This leads to the following equation for the power required to open the access transistors while writing or reading:

$$P_{\text{T,write}} = N_{BL} \frac{1}{2} C_{\text{gate}} V_{\text{WL,write}}^2 \cdot f \tag{3.12}$$

$$P_{\text{T,read}} = N_{BL}\frac{1}{2}C_{\text{gate}}V_{\text{WL,read}}^2 \cdot f \tag{3.13}$$

where C_{gate} is the gate capacitance of each transistor.

3.2.4 Total Power Consumption

The total write power can now be calculated by summing the contributions from equations (3.6), (3.8), (3.10) and (3.12), while the total read power can be estimated by adding the equations (3.7), (3.9), (3.11) and (3.13). This leads to:

$$P_{\text{write}} = P_{\text{WL,write}} + P_{\text{BL,write}} + P_{\text{cell,write}} + P_{\text{T,write}} \tag{3.14}$$

$$P_{\text{read}} = P_{\text{WL,read}} + P_{\text{BL,read}} + P_{\text{cell,read}} + P_{\text{T,read}} \tag{3.15}$$

Equations (3.14) and (3.15) forms the basic for estimating the power consumption, but they may have to be altered some after how each memory cell is designed and what is applicable. How this is done will be described in the section for each memory technology. In these sections, the estimations of $E_{\text{cell-write/bit}}$ and $E_{\text{cell-read/bit}}$, will be presented. As seen in equation (3.14) and (3.15), $E_{\text{cell-write/bit}}$ and $E_{\text{cell-read/bit}}$ represents the write and read energy per bit, without taking the charging of the interconnects or the switching of the access transistors into account.

3.3 Estimating Capacitance and Resistance of the Interconnects

As stated in equations (3.6), (3.7), (3.8) and (3.9), the capacitance of the interconnects plays an important role in the total energy consumption.

The model we used to estimate the capacitance in my earlier work [1] is shown in figure 3.1 [77]. The figure shows a cross-section of central conductor with two neighbouring conductors on each side, the conductors above and below are modelled as leading planes. The interconnects in the figure, 3.1, have a length L perpendicular to the cross-section.

With the following assumptions about the physical dimensions [78]:

- $w = s = F$, the metal half pitch or technology node. The spacing and thickness are equal to the half metal pitch.
- $t = h = A/R \cdot w$, where A/R is the aspect ratio. The height of each conducting layer is equal to the width times the aspect ratio used in the manufacturing process.

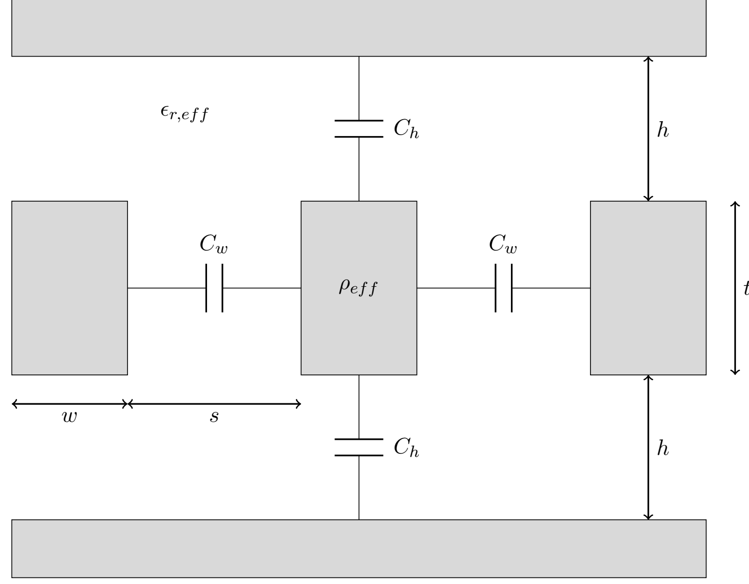


Figure 3.1: Model for estimating the capacitance and resistance of interconnects. This figure shows a cross-section of the interconnects. The length of the conductors perpendicular to the cross-section is denoted L.

The capacitance and resistance of the lines were estimated as [1]:

$$C_{tot} = 2\epsilon_0\epsilon_{r,eff}L\left(A/R + \frac{1}{A/R}\right) \tag{3.16}$$

$$R = \rho_{eff}\frac{L}{F^2 \cdot A/R} \tag{3.17}$$

The model gave good results for the resistance, but the estimated capacitance values were to low for sub 100 nm feature sizes, as the model does not take into account the fringing capacitances or the contact resistances which becomes important at this length scale. One important result from the capacitance model however, is that the total capacitance is proportional to the length of the interconnects.

In this work we will use another model, based on the work done by Zhirnov et al. [38], to calculate the capacitance. They have used measurements of different DRAM capacitances in the range from 90 to 40 nanometres. The DRAM bit lines they estimated for had an area factor $X_{AF} = 4$ and spanned 128 word lines. They then extrapolated this data as a power function and got the following model for a 128 word line long bit line [38]:

$$C_{tot} \approx aF^k \tag{3.18}$$

where $a \approx 2.24$ and $k \approx 0.6$, and F and C_{tot} are measured in nanometres and femtofarads respectively. We will now generalize this model to be valid for cases when the interconnect spans another number of bit lines, or have a different area factor than 4. This is needed as the capacitance of the interconnects will be longer if they have to span another number of word lines, or if cells themselves are larger as is given by a larger area factor. The length of the interconnects will scale proportional with the number of word lines, this was also shown by Zhirnov et al. [38]. If we further assume that cells are square, the length will be proportional with square-root of the area factor. This gives us the following generalized model:

$$C_{tot} \approx \frac{N_{WL}}{128}\sqrt{\frac{X_{AF}}{4}} \cdot aF^k \tag{3.19}$$

where the a and k is as before, 2.24 and 0.6 respectively. N_{WL} is the number of word lines and X_{AF} is the area factor of the memory cell. This model can also be used to estimate the word line capacitance; in this case N_{BL} is used instead of N_{WL}, where N_{BL} is the number of bit lines the word line has to span.

The parameters for the resistance model, developed in my earlier work [1], is extracted from the interconnect tables from ITRS interconnect chapter and tabulated in tables 3.3 and 3.4. Where the ITRS operates with a range of values, the average value has been chosen. Since some of the technology nodes is not present in the newest version of ITRS, older versions of the ITRS have been used as a source where necessary.

	Technology node F [nm]	Conductivity ρ_{eff} [µΩ cm]	Aspect ratio A/R	Source:
DRAM	31	2.2	1.8	[79]
SRAM	32	2.2	1.8	[79]
FeRAM	180	3.3	1.5	[80]
MRAM	90	3.3	1.7	[81]
STT-MRAM	65	2.2	1.7	[82]
PCRAM	38	2.2	1.8	[79]

Table 3.3: Interconnect resistance parameters 2012.

3.4 DRAM

The model for the DRAM was developed in my earlier work [1], but all parameters has been re-evaluated and updated to the newest version of the ITRS roadmap [14]. The DRAM retention model has been changed in this work, in order to take

	Technology node F [nm]	Conductivity ρ_{eff} [µΩ cm]	Aspect ratio A/R	Source:
DRAM	18	2.2	2	[79]
SRAM	16.9	2.2	2	[79]
FeRAM	90	3.3	1.7	[81]
MRAM	65	2.2	1.7	[82]
STT-MRAM	32	2.2	1.8	[79]
PCRAM	18	2.2	2	[79]

Table 3.4: Interconnect resistance parameters 2017.

into account the energy required to switch the gate capacitance of the access transistors.

The DRAM cell write energy per bit is estimated as charging a capacitor [9]:

$$E_{\text{cell-write/bit}} = \frac{1}{2} C_{\text{cell}} V_{\text{cell}}^2 \tag{3.20}$$

Where C_{cell} is the capacitance of the DRAM cell, and V_{cell} is the voltage swing over each cell, the values are collected from [14] and listed in table 3.5. In this model, it is assumed that the capacitor is empty and then charged to $\pm V_{\text{cell}}$, depending on whether a one or a zero is to be written.

	2012	**2017**
C_{cell} [fF]	25	25
V_{cell} [V]	0.55	0.48
$E_{\text{cell-write/bit}}$ [fJ]	3.8	2.9
V_{BL} [V]	0.55	0.48

Table 3.5: Parameters for estimating DRAM cell write energy per bit and bit line voltage. Adapted from: [14]

The cell write and read power can now be found from equation (3.10) and (3.11). It is further assumed that the bit line voltage is the same as V_{cell}, since the bit line capacitance and cell capacitance are in parallel. The values are listed in table 3.5. Writing a one or a zero costs the same amount of energy, because the only difference between writing a one or writing a zero is a change in the polarity of the voltage.

When it comes to reading, one only applies a voltage on the word line in order to open the transistor. However, the readout is destructive, so one has to add the writing energy in order to write-back the bit.

3.4.1 DRAM Refresh

The DRAM refresh period, τ_{ref}, is expected to be constant at 64 ms throughout the whole ITRS roadmap (i.e. to 2026) [14]. The refresh operation can be done in many different ways [83]. The assumption made here is that one has to read all the cells in each row every 64 ms, and since a read operation is followed by an immediate write-back this will refresh the memory state.

The power required to refresh the memory increases with the amount of cells needed to be refreshed [15]. Assuming N_{WL} is the number of word lines, and N_{BL} is the number of bit lines, the N_{WL} number of word lines has to be charged every τ_{refresh}. Then for each word line, N_{BL} number of bit lines and transistors has to be charged in order to write all the N_{BL} cells on each word line. This leads to the following equation:

$$\begin{aligned} P_{\text{ref}} =& \frac{N_{WL}}{\tau_{\text{ref}}} \left(\frac{1}{2} C_{WL} V_{WL,\text{write}}^2 \right. \\ & \left. + N_{BL} \cdot \left[\frac{1}{2} C_{BL} V_{BL,\text{write}}^2 + \frac{1}{2} C_{gate} V_{WL,\text{write}}^2 + E_{\text{cell-write/bit}} \right] \right) \end{aligned} \tag{3.21}$$

3.5 SRAM

The SRAM cell consists of six transistors, as shown in figure 2.18 on page 40. As mentioned in the MOSFET part of the theory section there exists multiple transistor designs in the ITRS roadmap. The transistor design is split into three major design paths. The transistors could either be designed for High Performance (HP), Low Operating Power (LOP) or Low Standby Power (LSTP) as discussed in section 2.3.10. For this model low standby power transistors are chosen as these will have the lowest leakage currents, as seen table 2.4.

In the coming years, the transistors in the roadmap are in a transition period in between extended planar bulk transistors, to fully depleted silicon on insulator (FD SOI) transistors or multi-gate transistors, like the FinFET [10]. It is here assumed that extended planar bulk transistors are used for SRAMs with gate length longer than 22 nm, while multi-gate transistors are assumed for transistors with gate lengths lower or equal to 22 nm.

As in the PIDS chapter in the ITRS [14], the dynamic power of a transistor is approximated as charging the gate capacitance for each transistor. In the ITRS, the gate capacitance C_{gate} is given as gate capacitance per gate width. The gate width is a parameter which is tuned for SRAM devices in order to suppress threshold voltage variations and achieve the maximum yield. A device width equal to three

times the gate length is assumed here i.e. $W_g = 3L_g$. This is in good accordance with [84] and [25], and should give a sufficient yield. Values down to $2L_g$ are suggested for some designs [25], but then the SNM can become to low, leading to reduced yield.

The condition $W_g = 3L_g$ is chosen for the two nMOS transistors in T_4 and T_6 in figure 2.18. It is important to size all the transistors correctly to get a good trade-off between read static noise margin and write noise margin. A cell ratio, CR, and pull-up ratio, PR, of 1 and 2 respectively is a good compromise, as described in the theory section. It is further assumed that the gate lengths of all the transistors in the cell are equal [25]. Inserting these assumptions into equations (2.38) and (2.40), we get the following relations for the different transistors widths:

$$W_{g,1} = W_{g,2} = W_{g,4} = W_{g,6} = 3L_g \tag{3.22}$$

$$W_{g,3} = W_{g,5} = 6L_g \tag{3.23}$$

where $W_{g,i}$ are the gate width of the corresponding transistor in figure 2.18. Assuming an equal amounts of ones and zeros are written, half of the time the cell does not have to switch its state. The cell write energy per bit for the four transistors making up the storage element can now be estimated as:

$$E_{\text{cell-write/bit}} = \frac{1}{2}\left(\frac{1}{2}C_{\text{gate}}V^2\left[3L_g + 3L_g + 6L_g + 6L_g\right]\right) = \frac{9}{2}C_{\text{gate}}V^2 \cdot L_g \tag{3.24}$$

For a read operation, the access transistors are opened, and the stored state is driven into the complementary bit lines, i.e. there is no energy required to switch any states in the cell. However, it still costs energy to charge the capacitance of the interconnects and the access transistors, but this is accounted for by another part of the model, as stated in equation (3.15).

The SRAM cell has two bit lines per cell, so this is accounted for by multiplying the bit line capacitance by two. The read and write voltages are collected from [14].

The SRAM needs to have constantly applied power in order to store the data in the cells. The leakage current is collected from [14] as the $I_{\text{sd-leak}}$, which is defined as the source drain leakage per unit gate length for a nMOS transistor. It is assumed that the source drain leakage is larger than the gate and junction leakage currents at room temperature [10, 14]. For the 6T SRAM cells there are two paths where the current can flow, and two nMOS transistors for each pair. With the assumptions from equation (3.22), the static power required to retain the data per cell can be estimated as:

$$P_{\text{static}} = 2V_{dd}I_{\text{sd-leak}} \cdot (L_g + L_g) = 4V_{dd}I_{\text{sd-leak}} \cdot L_g \tag{3.25}$$

	2012	**2017**
V [V]	0.9	0.75
C_{gate} [fF µm^{-1}]	0.866	0.567
$C_{\text{gate}}V^2$ [fJ µm^{-1}]	0.7	0.32
L_g [nm]	27	15.7
$E_{\text{cell-write/bit}}$ [fJ]	0.085	0.023
V_{BL} [V]	0.9	0.75
$I_{\text{sd-leak}}$ [pA µm^{-1}]	10	10
P_{static} [pW/cell]	1.458	0.707

Table 3.6: Parameters for estimating SRAM cell write energy per bit and the bit line voltages. Adapted from: [14].

3.6 FeRAM

The model for the FeRAM was developed in my earlier work [1]. Here, all the values has been re-evaluated and updated to the newest version of the roadmap [14].

Similar to the DRAM, the write energy per bit for the FeRAM is estimated as switching the capacitor. The energy required to switch the ferroelectric capacitor is estimated as [9]:

$$E_{\text{cell-write/bit}} = \frac{1}{2}\sigma A_{\text{act}} V \tag{3.26}$$

where σ is the minimum switching charge density, A_{act} is the active area of the capacitor and V is the voltage over the ferroelectric capacitor. The values are collected from [14] and listed in table 3.7. This model assumes that the memory is switched every write operation.

The FeRAM cell also uses a drive line to read and write, so that has to be incorporated into the power calculations as well. It is assumed that the drive line capacitance is the same as the bit line capacitance, since it is processed with the same parameters, and has to be of similar length. It is further assumed in our model, that the voltage over the bit line and drive line is equal to the operating voltage listed in table 3.7. It is also assumed that an equal amount of ones and zeros are written. When writing a one, a voltage is applied on the bit line, plate line and word line. The difference between writing a one and a zero is that when writing a zero, no voltage is applied to the bit line, see figure 2.22. Just as the DRAM, the FeRAM cell has a destructible readout, so the write energy is added to the read energy to account for that.

The cell write and read power can now be found from equation (3.10) and (3.11).

	2012	**2017**
σ [µC µm^{-2}]	8.5	12
A_{act} [µm^2]	0.423	0.234
V [V]	1.5	1.2
$E_{\text{cell-write/bit}}$ [fJ]	33	15

Table 3.7: Parameters for estimating FeRAM cell write energy per bit. Adapted from: [14].

3.7 MRAM and STT-MRAM

Both the conventional field switching MRAM and the STT-MRAM are described in this section as the procedures to calculate the write and read energies are similar; however, the parameters often deviate. The STT-MRAM model was developed in my earlier work [1] and all the parameters has been re-evaluated and updated. It has also been updated to take into account the energy consumed in the parasitic FET channel resistance. The model to estimate the energy for field switching MRAM has been developed in this work.

The physical mechanisms of writing and reading the MRAM and STT-MRAM are different, as described in the theory section. Still, the quantity that is expressed in the ITRS roadmap for the write energy $E_{\text{cell-write/bit}}$, is calculated as write current times write voltage times write time, i.e. $I_{\text{write}} \cdot V_{\text{write}} \cdot t_{\text{write}}$ [14]. This means that although the mechanisms are different, by looking at the applied currents and voltages when writing, one can estimate the power without considering if it is field-switching or spin-torque-transfer switching. It is shown in [14] that the $E_{\text{cell-write/bit}}$ is 120 pJ and 2.2 pJ in 2012 and 110 pJ and 0.3 pJ in 2017, for conventional MRAM and STT-MRAM respectively.

Here, we further assume that the field switching MRAM is implemented as a toggle MRAM. As discussed in theory section, toggle MRAM cells must be sensed before they are read. To account for this, the read energy is added to the write energy. Assuming an equal amount of zeros and ones are written, this means that we need to toggle the MRAM half of the time. The MRAM has a third line, often called the digit line, and it is assumed that this line has the same length and dimensions as the bit line, meaning it will have the same capacitance. In contrast to the other technologies, when switching the conventional MRAM cell no voltage is applied to the word line. Meaning that it does not need to be charged, and the gate capacitance of the access transistors does not have to be switched. However, the word line and the gate capacitances still have to be switched in the read operation that happens in advance of a toggle operation.

We also need to estimate the cell read energy per bit. The equivalent circuit of a MRAM or STT-MRAM cell while reading is shown in figure 2.29, on page 56. When reading, a voltage V_{read} is applied to the bit line, the resistance of the cell is either R_P or R_{AP} depending on whether a zero or one was stored. Assuming an equal amount of ones and zeros are read and including the resistance of the bit line, the energy required to read the cell can be estimated as [38]:

$$E_{\text{cell-read/bit}} = \frac{V_{\text{read}}^2 t_{\text{read}}}{2} \left(\frac{1}{R_P + R_{BL} + R_{FET}} + \frac{1}{R_{AP} + R_{BL} + R_{FET}} \right) \tag{3.27}$$

where V_{read} is the read voltage R_P and R_{AP} is the resistance of parallel and anti-parallel states respectively. R_{BL} is the bit line resistance, R_{FET} is the transistor channel resistance and t_{read} is the read time. This formula can be used for both conventional MRAM and STT-MRAM, as the difference between them are in writing mechanisms, not reading mechanisms. The read energy per bit should be lower than the write energy per bit. The R_P is found by dividing resistance area product, $R \cdot A_{\text{act}}$, listed in [14], by the active area, A_{act}. The R_{AP} can then be calculated through the TMR ratio, by equation (2.45) on page 51. The optimal read voltage is between 200 and 300 mV [85], the average value is chosen in the calculations. The read time for 2012 is assumed to be 10 ns [86] and 1 ns in 2017 [86, 87]. These values are tabulated in tables 3.8 and 3.9.

	2012	**2017**	Sources
$E_{\text{cell-write/bit}}$ [pJ]	120	110	[14]
$R \cdot A_{\text{act}}$ [$\Omega\,\mu\text{m}^2$]	1200	600	[14]
A_{act} [μm^2]	0.124	0.066	[14]
TMR [%]	65	90	[14]
R_P [kΩ]	9.677	9.090	Calculated
R_{AP} [kΩ]	15.97	17.27	Calculated
V_{read} [mV]	250	250	[85]
t_{read} [ns]	10	1	[86, 87]
$V_{BL,\text{write}}$ [V]	1.8	1.5	[9, 88]

Table 3.8: Parameters for estimating conventional field switching MRAM cell write and read energy per bit, as well as the bit line voltages.

The cell write and read power can now be found from equation (3.10) and (3.11). It is further assumed that the bit line voltage during writing is 1.8 V in 2012 [9] and 1.5 V in 2017 [88]. While reading the bit line voltage is assumed to be 250 mV [85]. The values are also tabulated in tables 3.8 and 3.9.

	2012	**2017**	Sources
$E_{\text{cell-write/bit}}$ [pJ]	2.2	0.3	[14]
$R \cdot A_{\text{act}}$ [$\Omega\,\mu\text{m}^2$]	11	10	[14]
A_{act} [μm^2]	0.008	0.003	[14]
TMR [%]	120	150	[14]
R_P [kΩ]	1.375	3.333	Calculated
R_{AP} [kΩ]	3.025	8.333	Calculated
V_{read} [mV]	250	250	[85]
t_{read} [ns]	10	1	[86, 87]
$V_{BL,\text{write}}$ [V]	1.8	1.5	[9, 88]

Table 3.9: Parameters for estimating STT-MRAM cell write and read energy per bit, as well as the bit line voltages.

3.8 PCRAM

A new model to estimate the energy consumption for the Phase Change RAM is developed in this section. The reason why a new model is needed, is because the way the set current was estimated in my earlier work [1] was too simplified, and in turn gave too low energy consumption. Another flaw considering the old model was that it did not take into account the effects of write and verify operations which are needed for multi-level cells (MLC).

3.8.1 Single-Level Cells (SLC)

We will start our methodology with the single-level cell (SLC), and we will later generalize these results to the multi-level situation. The Phase Change RAM is different compared to the other types of memories, because it requires a very different amount of energy to reset and set a bit. If we have SLC, and assume that an equal amount of ones and zeros are written, the write energy per bit can be calculated as:

$$E_{\text{cell-write/bit-SLC}} = \frac{E_{\text{reset}} + E_{\text{set}}}{2} \tag{3.28}$$

where E_{reset} and E_{set} is the energy required to completely reset or set a cell respectively. A typical IV curve of a phase change material is shown in figure 3.2. It shows that the high voltage resistance, which is observed during switching, is in the same order as the set resistance, R_{set} [66, 89].

From this we can estimate the energy cost for fully resetting a bit for PCRAM, it

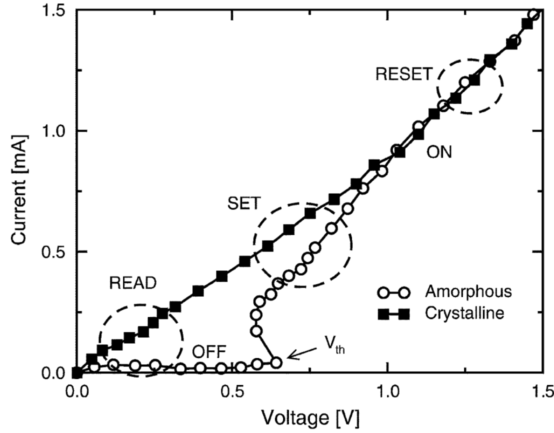


Figure 3.2: Typical IV curve for phase change materials. From [89].

is calculated as [9]:

$$E_{\text{reset}} = I_{\text{reset}}^2 \left(R_{\text{set}} + R_{BL} + R_{FET}\right) t_{\text{reset}} \tag{3.29}$$

where the I_{reset} is the current required to reset the cell and t_{reset} is the length of the current pulse. The R_{BL} is the resistance of the bit line and R_{FET} is the resistance of the access transistor channel. The reset current and set resistance is found in [14], the reset time is collected from [64, 68]. The values are listed in table 3.10.

Similar to the reset, the energy cost for a full set operation is estimated as:

$$E_{\text{set}} = I_{\text{set}}^2 \left(R_{\text{set}} + R_{BL} + R_{SET}\right) t_{\text{set}} \tag{3.30}$$

The set time is estimated from [9], which states that the set time should be 100 ns in 2011 and below 50 ns in 2024. A linear decrease in t_{set} from 100 ns in 2011 to 50 ns in 2024 is then assumed.

The reset current is given in the ITRS roadmap, but not the set current. Therefore, it needs to be estimated, and it is done as follows: The change in temperature of the phase change material is proportional to the power delivered; the power delivered is again proportional to the square of the current, i.e. [90]:

$$\Delta T \propto I^2 \tag{3.31}$$

where ΔT is the change in temperature and I is the applied current. We assume that the phase change material needs to be heated by 600 °C for a reset operation, but only 200 °C for a set operation. We find that the set current can be

	2012	**2017**	Sources
I_{reset} [µA]	174	57	[14]
I_{set} [µA]	100	33	Calculated
t_{reset} [ns]	10	6	[14]
t_{set} [ns]	95	75	[9, 64]
R_{reset} [kΩ]	300	1000	[64, 68]
R_{set} [kΩ]	2.6	5.5	[14]
V_{read} [V]	0.2	0.2	[91]
t_{read} [ns]	12	2	[92, 93]

Table 3.10: Parameters for estimating PCRAM write and read energy per bit, as well as the bit line voltages.

approximated with:

$$I_{\text{set}} \approx \sqrt{\frac{200\,°\text{C}}{600\,°\text{C}}} I_{\text{reset}} = \frac{1}{\sqrt{3}} I_{\text{reset}} \approx 0.58 I_{\text{reset}} \tag{3.32}$$

This is good accordance with the data presented by Lee et al. [65], where the set current varies with $1/3 \sim 2/3$ of the reset current.

We can now estimate the voltage which needs to be applied to the bit line, in order to achieve these currents as:

$$V_{BL,\text{write}} = \begin{cases} I_{\text{reset}} \left(R_{BL} + R_{FET} + R_{\text{set}}\right) = V_{BL}^{(0)} & \text{for writing zero} \\ I_{\text{set}} \left(R_{BL} + R_{FET} + R_{\text{set}}\right) = V_{BL}^{(1)} & \text{for writing one} \end{cases} \tag{3.33}$$

from the same argument as for the reset and set energies, the R_{set} is used as an approximation for the high field resistance.

Similar to equation (3.27), the read energy per bit for the SLC PCRAM can be estimated as:

$$E_{\text{cell-read/bit-SLC}} = \frac{V_{\text{read}}^2 t_{\text{read}}}{2} \Bigg(\frac{1}{R_{\text{set}} + R_{BL} + R_{FET}} + \frac{1}{R_{\text{reset}} + R_{BL} + R_{FET}} \Bigg) \tag{3.34}$$

The read voltage is assumed to be 0.2 V in 2012 [91] and the read time in 2012 is assumed to be 12 ns [92]. The reset resistance is collected from [64] for single level cells. According to [9] the read time should be below 10 ns in 2024, and 2 ns is chosen for 2017 [93]. The values are also tabulated in table 3.10.

3.8.2 Multi-Level Cells (MLC)

Because the ITRS assumes that four bits can be stored in each cell in 2017 [14] we need to estimate how this will affect the energy consumption. The energy consumption will be altered because write and verify programs are needed in order to make sure that the stored resistance is within a given bandwidth [68, 69].

We will start by developing a model for the different target resistance levels. The first assumption we then make, is that the target resistance levels are between R_{set}, given by the ITRS [14] and up to full reset, which is assumed to be $R_{\text{reset}} = 1\,\text{M}\Omega$ [68]. The intermediate resistance levels are then assumed to be equally spaced on a logarithmic scale, as they are in figure 2.35b. This is a common way to space the resistance levels, in order to work towards non-overlapping resistance distributions [11]. With n resistance levels one can store $\log_2(n)$ bits and the resistance of the i-th level can be calculated as:

$$\log(R_i) = \log(R_{\text{set}}) + \frac{i-1}{n-1} \cdot \Big[\log(R_{\text{reset}}) - \log(R_{\text{set}})\Big] \tag{3.35}$$

where log denotes the logarithm with base 10. Note that this formula gives $R_1 = R_{\text{set}}$ and $R_n = R_{\text{reset}}$.

There exist many approaches with different combinations of pulses and ramp down time in order to partially write a cell [67–70, 94–100]. They all have two things in common: The first is that all of them use a write and verify algorithm. The other one is that they vary the amplitude and/or the length of the pulses, to achieve the target resistance.

To develop a model for the energy consumption we will assume that the stair-case up (SCU) algorithm is utilized for writing [69, 96, 97, 100]. The SCU method is a partial set write and verify algorithm, and it is sketched in figure 3.3. As seen in the figure, the operation starts with a reset of the cell, and then multiple write operations are done with increasing amplitude of the voltage (which corresponds to increasing current amplitude). This is repeated until the target resistance is achieved. The stair-case up algorithm is simple, as the same pulse width is used in all the pulses, this makes it easier to implement in hardware [69]. The SCU algorithm is also one of the best documented algorithms, which makes it easier to justify any assumptions made.

As the SCU algorithm is a partial set method it is considered more immune to resistance drift, because the crystalline phase is more stable than the amorphous phase [69]. However, with this algorithm the resistance of each step can only be reduced as each write pulse can only increase the amount of crystalline material

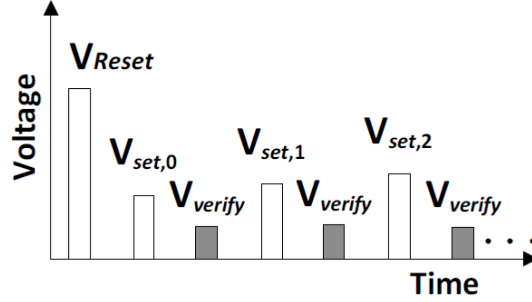


Figure 3.3: Conventional MLC PCRAM stair-case-up (SCU) write algorithm. The cell is first reset and then written with different voltages until target resistance is achieved. After each write a read operation is performed, indicated by the grey amplitudes. Note that the applied write voltage amplitude increases for each iteration. Initial set pulse not shown. From [69].

[99]. The only way to increase the resistance again is to start over with a new reset pulse [96, 99].

It is considered standard to read the cell first to check if a read operation is necessary, this is done to prevent unnecessary writes and extend the endurance [69]. Assuming that the data is coded so that all n resistance values have the same probability, the chance of not needing to write the cell is then $1/n$.

It is also considered standard to first completely set the cell with a set pulse or sweep before the writing algorithm start, as sketched in figure 3.3. This is done to avoid void formations and get a more predictable write result [96, 97]. Further, if a write operation is to write R_{set} or R_{reset} it is sufficient with the first set or reset pulse respectively in order to achieve the target resistance [67, 96]. With the same arguments as before, the chances of writing R_{set} or R_{reset} is $1/n$ for each case.

The writing sequence is summarized in figure 3.4, using this figure as a reference and the fact that we are writing $\log_2(n)$ bits at a time, we have that the write energy per bit is given by:

$$
\begin{aligned}
E_{\text{cell-write/bit-MLC}} = & \frac{1}{\log_2(n)}\Bigg(E_{\text{read-MLC}} + \frac{n-1}{n}\Bigg[E_{\text{set}} + \frac{n-1}{n}\Bigg(E_{\text{reset}} \\
& + \frac{n-2}{n}\Bigg[N \cdot E_{\text{read-MLC}} + \sum_{i=0}^{N-1} E_i\Bigg]\Bigg)\Bigg]\Bigg)
\end{aligned}
\tag{3.36}
$$

where $E_{\text{read-MLC}}$ is the energy required to read a multilevel cell, E_{set} and E_{reset} is the energy required for fully setting and resetting the cell respectively. This is

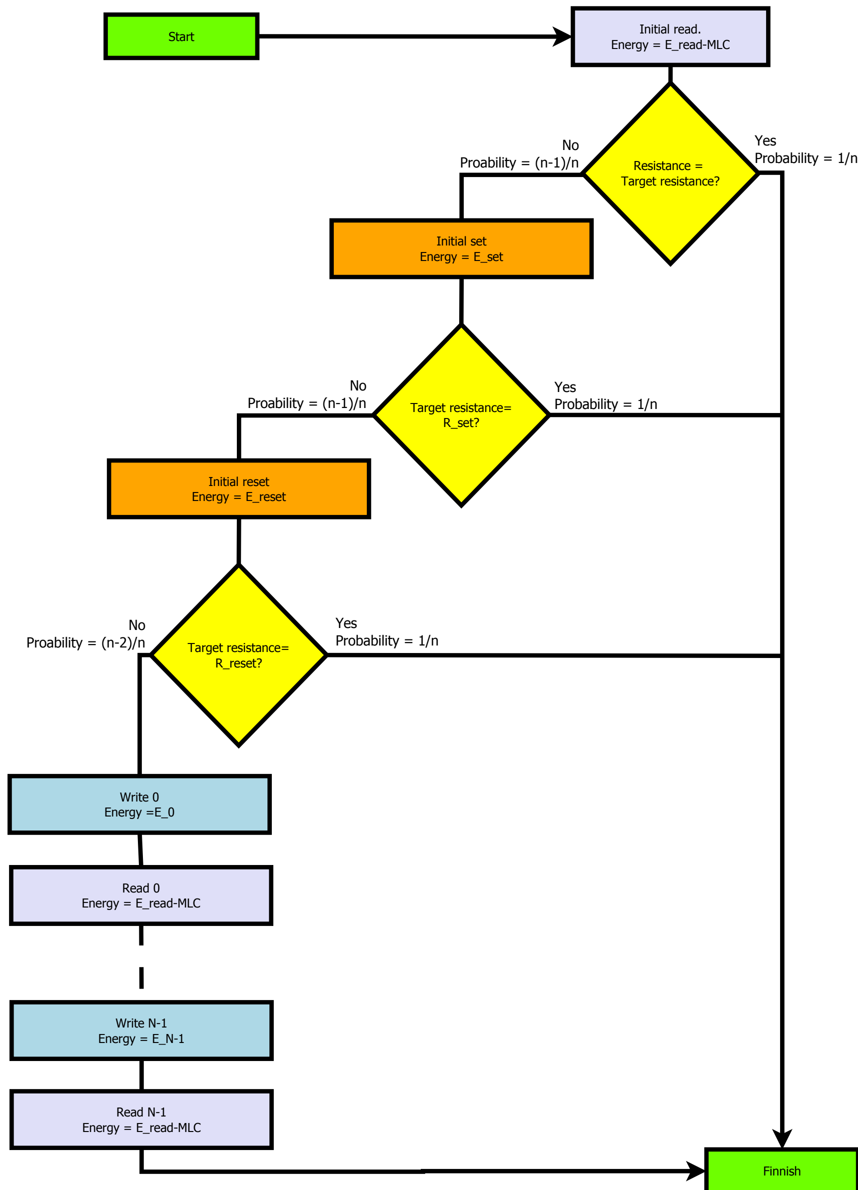


Figure 3.4: Flowchart showing the write sequence in a MLC PCRAM assuming N operations are needed to achieve the target resistance. The energy cost of each operation is indicated in the corresponding box. Based on [69, 96, 97].

the same as the set and reset energies as in a single-level cell, N is the required number of iterations in order to achieve the target resistance and E_i is the energy required for the i-th set operation.

We start by estimating the energy required for in the i-th write operation, E_i. It is assumed that each pulse has the same length as the reset pulse, t_{reset}. We further assume that current amplitude start as I_{set}, which is the amplitude needed to set the material for the single level approach, and increase by ΔI each iteration [96]. This leads to:

$$E_i = (R_{\text{set}} + R_{BL} + R_{FET}) \cdot t_{\text{reset}}(I_{\text{set}} + i \cdot \Delta I)^2 \tag{3.37}$$

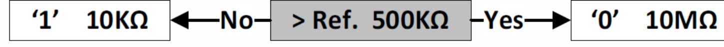
The increase for each step, ΔI, should be large in order to increase writing efficiency; however, if it is too large writing errors are more likely to occur [97]. This error can either be fixed by applying a reset pulse and start again from the beginning [96]. Or if it is a rare event, it can be left there and later handled by error correcting circuitry (ECC) [69]. Regardless, the step size, ΔI, is a trade-off between writing efficiency and writing accuracy [97,100]. The step size is therefore assumed to be [67]:

$$\Delta I = \frac{I_{\text{reset}} - I_{\text{set}}}{2N_{\text{average}}} \tag{3.38}$$

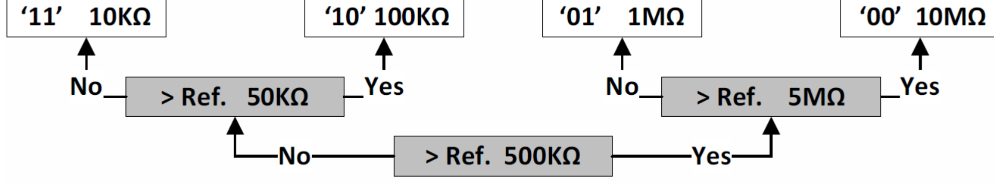
where N_{average} is the average number of iterations needed. This will cause current pulses to reach I_{reset} if $2N_{\text{average}}$ pulses are needed to achieve the target resistance. N_{average} for state of the art MLC PCRAM, with two bits per cell using the SCU algorithm are $2 \sim 3$ [69]; however, this value is expected to increase exponentially with the number of bits per cell [101]. Since the ITRS assumes that four bits per cell is used in 2017 [14], it is here assumed that $N_{\text{average}} = 2e^4/e^2 \approx 15$ in 2017. And since we are interested in the write energy per bit, it is assumed for simplicity that all write operations need N_{average} operations.

What is left now is to estimate the power consumption of the read operation for a multilevel cell. When reading a single-level cell, the resistance value is compared against a resistor ideally at the midpoint between R_{set} and R_{reset}, as illustrated in figure 3.5a. In a multi-level cell this is conventionally done with a binary search algorithm, as shown in figure 3.5b [69]. This means that $\log_2(n)$ number of reads must be done to sense the correct resistance (this is equal to the number of bits per cell). If the read time is critical, it is possible to do this sensing in parallel, in that case the sensing circuit must be copied $n-1$ times. This represents a trade-off between read performance and the extra sensing circuitry, and the conventional approach is to use sequential sensing [69].

The energy required to read a single cell with resistance R_i can be calculated



(a) SLC



(b) MLC

Figure 3.5: Binary search algorithm for PCRAM read. (a) Single-level cell (SLC). (b) Multi-level cell (MLC). Here with $n = 4$ i.e. two bits per cell. From [69].

similar to the read for a single level cell as in equation (3.34):

$$E_{\text{read},i} = \frac{V_{\text{read}}^2 t_{\text{read}}}{R_i + R_{BL} + R_{FET}} \tag{3.39}$$

where R_i is the resistance of level i, as given by equation (3.35), R_{BL} is the bit line resistance and R_{FET}, is the resistance of the FET channel. V_{read} and t_{read} is the read voltage and read time respectively. Assuming a sequential read algorithm, the average read energy can be estimated as $\log_2(n)$ times the average of $E_{\text{read},i}$, i.e.:

$$E_{\text{read-MLC}} = \frac{\log_2(n)}{n} \sum_{i=1}^{n} E_{\text{read},i} \tag{3.40}$$

We also have to take into account that a read operation reads $\log_2(n)$ bits at a time. Therefore, the cell read energy per bit for a multi-level cell is:

$$E_{\text{cell-read/bit-MLC}} = \frac{1}{n} \sum_{i=1}^{n} E_{\text{read},i} \tag{3.41}$$

We now have all we need to estimate the write and read energy per bit for the multi-level cell. So far we have not mentioned the cost to charge the capacitances of the bit lines and word lines and switching the access transistors. This will have to be done for each read and write operation which are done during the write and verify algorithm, and is estimated as for the other technologies.

3.9 Estimating the Power Associated with the Access Transistors

In order to model the energy consumed to open the access transistors, we use the equations (3.12) and (3.13). It is here assumed that all the access transistors are standard nMOS transistors. This might not be the case, as some technologies might require special access devices. However, it is preferable to use standard logic transistors as access device, because this reduces the need for special masks and extra processing steps [13, 92]. As there is no section describing the access devices for the different memory technologies in detail, in the ITRS roadmap, using the values for logic devices is done to get a good estimate.

Since some of the technology nodes are not present in the newest version of ITRS, older versions of the ITRS have been used as a source where necessary, as seen in tables 3.11 and 3.12.

3.9.1 Transistor Design Path

The first assumption we need to do, is to assume what kind of transistor design which are used for the memory technologies. As discussed in the theory section this is a trade-off between maximum operating frequency versus low operating power or low standby power. It is here assumed that transistors designed for low standby power are used for the volatile technologies, because this will give the lowest leakage currents. These transistors also have the largest I_{on}/I_{off} ratios, which are especially important for DRAM [13]. Transistors designed for low operating power are assumed for the non-volatile technologies. The exception from this is in case of STT-MRAM and PCRAM where high performance transistors are assumed, because HP transistors have the largest on current [14]. As discussed in the theory section, these memory technologies can be limited by the access device's ability to deliver current. Low operating power will be most important for non-volatile devices as these devices can be turned off when not used, so that large leakage currents are not a problem as for the volatile memories.

It is also assumed that extended planar bulk transistors are used for gate lengths above 22 nm, and that multi-gate transistors like the FinFET are used for gate lengths equal to or lower than 22 nm. An exception from this is DRAM, where multi-gate transistors are already in production [14]. Therefore, for DRAM multi-gate transistors are assumed in both 2012 and 2017. For the SRAM, two bit lines are needed so that double the amount of access transistors is needed.

3.9.2 Gate Capacitance and Channel Resistance

The gate capacitance is collected from the ITRS roadmap as the entry "C_g total gate capacitance" [14]. However, in some of the older versions of the roadmap, the gate capacitance is not listed explicitly. In these cases the gate capacitance is calculated with the same formula as it is done in the newer versions of the roadmap [102]:

$$C_g = \left(\frac{\epsilon_{SiO_2}}{T_{ox-el}} L_g + C_{\text{total fringing}} \right) W_g \tag{3.42}$$

where ϵ_{SiO_2} is the permittivity of silicon dioxide and $C_{\text{total fringing}}$ is the total fringing capacitance plus Miller effect. The Miller effect is a change in input capacitance because of the amplification process [103]. T_{ox-el} is the oxide electric thickness taking into account dark space and polydepletion effects, as described in the MOSFET theory; it is defined as [102]:

$$T_{ox-el} = t_{eq} + t_{\text{dark space}} + t_{\text{poly depletion}} \tag{3.43}$$

t_{eq} is defined in equation (2.36). Dark space is an increase in the actual oxide electric thickness, due to the formation of quantum states in the gate and channel. Polydepletion is the depletion of carriers at the polysilicon-oxide interface [104]. $t_{\text{dark space}}$ typical values are $2 \sim 4$ Å for electrons [102,104] and 3 Å is assumed here. A typical value for $t_{\text{poly depletion}}$ is 4 Å for nMOS transistors [102, 104]. A typical value for $C_{\text{total fringing}}$ is $0.24\,\text{fF}\,\mu\text{m}^{-1}$ [102].

The equivalent resistance of the MOSFET Channel is taken from the ITRS roadmap as the entry "R_{sd} - Effective parasitic series source/drain resistance" and it is given in $\Omega\,\mu\text{m}$. This means that the equivalent resistance of the channel for a given gate width W_g can be calculated as:

$$R_{FET} = \frac{R_{sd}}{W_g} \tag{3.44}$$

3.9.3 Word Line Voltages

As discussed in the theory section, the voltage applied to the word line is the voltage needed for making the transistor channel conducting. Therefore, the word line voltage assumed to be equal to the supply voltage, V_{DD}. There are two exceptions from this: One is for the normal MRAM, which only needs a voltage on the word line for reading operation. The other exception assumption is DRAM, where highly boosted V_{DD} and elevated threshold voltages are used in order to suppress leakage currents [13,14]. To account for this, the DRAM word line voltage

is taken from the DRAM tables in the PIDS chapter as "maximum word line level" [14].

3.9.4 Estimating the Gate Width

As mentioned in the SRAM methodology, transistor power consumption depends on the gate width. For SRAM it is assumed that the gate width is three times the gate length [25, 84]. For the other technologies, where the sizing of the transistor does not have the same impact on the yield, it is assumed that the gate width is two times the gate length [14, 25]. This gate width is the equivalent channel width as if the transistor is planar, thus it can be a height in case of 3D transistors such as FinFET. Two exceptions to the $W_g = 2L_g$ assumption are for STT-MRAM and PCRAM, because of the large currents which are needed in order to switch the cells. As the on current of a transistor depends on the gate width, they need transistors with a large enough gate width in order to be able deliver enough current [14]. This width is calculated as [14]:

$$W_g = \frac{I_{\text{switch}}}{I_{\text{d-sat per gate width}}} \tag{3.45}$$

where I_{switch} is the largest required current to switch the cell and $I_{\text{d-sat per gate width}}$ is the drain current per gate width for high performance devices, corresponding to the feature size. The switching current for STT-MRAM is 175 µA and 50 µA in 2012 and 2017 respectively [14]. While for the STT-MRAM transistors $I_{\text{d-sat per gate width}}$ is 1006 µA µm^{-1} and 1469 µA µm^{-1} in 2012 and 2017 [14, 105]. For PCRAM, the largest current is the reset current which is 224 µA and 57 µA in 2012 and 2017 respectively [14]. The $I_{\text{d-sat per gate width}}$ for PCRAM transistors is 1320 µA µm^{-1} in 2012 and 1744 µA µm^{-1} and 2017 [14]. However, if this calculated gate width is lower than two times the channel length, the gate width is assumed to be two times the channel length instead.

A summary of the important parameters for the access transistors are given in tables 3.11 and 3.12.

3.10 Further Assumptions

The powers consumed by the memories are calculated by equation (3.14) and (3.15). First, before the consumed power is calculated, we need to estimate the capacitance and resistance of the interconnects. These are given by equations (3.19) and (3.17). The resistance and capacitance are proportional to the length

	F [nm]	L_g[nm]	V_{WL} [V]	C_g [fF µm^{-1}]	R_{sd} [Ω µm]	W_g[nm]	Source:
DRAM	31	27	2.7	0.669	467	54	[14]
SRAM	32	27	0.9	0.866	290	81	[14]
FeRAM	180	120	1.5	1.669	200	240	[106]
MRAM	90	53	0.9	1.036	180	106	[107]
STT-MRAM	65	29	1.1	0.721	200	174	[105]
PCRAM	38	24	0.9	0.936	330	170	[14]

Table 3.11: Access transistor parameters 2012.

	F [nm]	L_g[nm]	V_{WL} [V]	C_g [fF µm^{-1}]	R_{sd} [Ω µm]	W_g[nm]	Source:
DRAM	18	15.7	2.4	0.567	264	31.4	[14]
SRAM	16.9	15.7	0.75	0.567	264	47.1	[14]
FeRAM	90	53	0.9	1.036	180	106	[107]
MRAM	65	32	0.8	0.789	190	64	[105]
STT-MRAM	32	22	0.87	0.752	366	44	[14]
PCRAM	18	14	0.75	0.611	235	33	[14]

Table 3.12: Access transistor parameters 2017.

of the interconnects, as seen in equation (3.19) and (3.17). Assuming a size on the memory matrix has to be done in order to get an estimate for the length needed for the interconnects. The following memory size is assumed: 32 kB, because it is a typical memory block size in a microcontroller [2,13,108]. To achieve a memory size of 32 kB, the following number of cells is needed:

$$N_{\text{cells}} = \frac{32\text{kB}}{\text{bits/cell}} = \frac{32 \cdot 1024 \cdot 8 \text{ bits}}{\text{bits/cell}} \tag{3.46}$$

Assuming a square matrix, the number of bit lines and word lines needed are:

$$N_{BL} = N_{WL} = \sqrt{N_{\text{cells}}} \tag{3.47}$$

Further, assuming that the cells are square, the minimum length needed for the interconnects to span the whole matrix is:

$$L = N_{BL}\sqrt{A} \tag{3.48}$$

where A is the area of the cell, given by equation (3.1).

For the power, given by equations (3.14) and (3.15), the clock frequency f and the word size is also needed. It is assumed that the frequency is 32 MHz, because this is a typical microcontroller clock frequency [2, 109]. The word size N_{ws} is generally a multiple of 8 because that is the size of one byte, $N_{ws} = 16$ is assumed here.

3.11 Demands for the Sense Amplifiers

After a cell has been read, it has to be converted to digital information by a sense amplifier. The energy consumed by the sense amplifier is not taken into account by our model. All of the technologies needs sense amplifiers; however, the sense amplifier design depends on the signals which are to be detected. A smaller difference between a logic one and a logic zero requires a more complex sense amplifier [13]. The sense amplifier varies for different realizations of the same memory technology, as it is generally made with other design criteria than the rest of the device to minimize the device errors [33]. Still, we have developed a methodology to compare the demands for the sense amplifiers.

We will here try to give an estimate of the read signal strengths. If the signal to be detected is a voltage, as for DRAM, SRAM and FeRAM, the relative signal magnitude between a logic one and a logic zero is estimated as:

$$S = \frac{V^{(1)} - V^{(0)}}{V^{(1)}} \tag{3.49}$$

where the $V^{(1)}$ and $V^{(0)}$ is the voltage if a logic one and a logic zero is stored respectively. Similar, the signal strength if a current is to be sensed, as is the case for the resistive memories the relative signal strength is estimated as:

$$S = \frac{I^{(1)} - I^{(0)}}{I^{(1)}} \tag{3.50}$$

where the $I^{(1)}$ and $I^{(0)}$ is the current if a logic one and a logic zero is stored respectively. This will make it possible to compare the different signals strength regardless if it is a current or a voltage which is to be sensed. Assuming that the current or voltage sensed is higher for a logic one than a logic zero, the relative signal strength will be a quantity in the interval $0 < S \leq 1$, and a signal strength of $S = 1$ represents the best possible value. In the following, we will give an explanation for how these signal strengths are estimated.

For DRAM, the signals $V^{(1)}$ and $V^{(0)}$ are estimated with equation (2.34), with $V_{DD} = 2V_{cell}$.

For the SRAM, the signal strength is determined by the magnitude of the differential signal that develops on the complementary bit lines when they are discharged through the cell. This differential signal depends on the design of the driver circuit and the sense amplifier [110]. It is important that the differential signal amplitudes does not exceed the threshold voltage of the transistors, as this can destroy the stored state (the readout becomes destructive) [25]. So to get an upper limit for the signal strength it is assumed that $V^{(1)} - V^{(0)} = V_{th}$ and that $V_1 = V_{dd}$ [33]. The threshold voltage for the low standby power transistors that are assumed in this work are 637 mV and 479 mV in 2012 and 2017 respectively [14].

For FeRAM, the signals $V^{(1)}$ and $V^{(0)}$ are estimated with equation (2.42) with A as the active area and P_{sw} as the minimum switching charge, given by the ITRS. In the non switching case ΔP_{nsw} can be found by looking at the ferroelectric film as a capacitor [111]. The non-switching charge can now be estimated as:

$$\Delta P_{nsw} \approx \epsilon_0 \epsilon_r \frac{V}{d} \tag{3.51}$$

We further assume that the thickness of the ferroelectric capacitor is equal the height of one metal layer $d \approx A/R \cdot F$, where A/R is the aspect ratio and $\epsilon_r \approx 1000$ [112] for PZT.

For STT-MRAM and MRAM the read currents $I^{(1)}$ and $I^{(0)}$ can be estimated with equation (2.48).

The PCRAM read currents, $I^{(1)}$ and $I^{(0)}$, can be estimated with equation (2.52). In the case of multi-level storage it is assumed that sequential sensing is utilized and that the resistances, calculated with equation (3.35), which gives the lowest signal strength are used as R_{set} and R_{reset} in equation (2.52).

3.12 Iso-Feature Size Estimates

One issue when comparing the results, which are based on the predictions for the memory technologies from the ITRS, is that for a given year the feature size of the different technologies varies significantly. This is not the situation when considering an embedded memory technology for a given application. For an embedded system, the feature size of the memory will most likely be similar to that on the rest of the chip [2].

In order to get more insights in the differences between the memory technologies, we will here look at the situation if the feature size of the memories are made similar to each other. To do this we need to select a feature size which is represented for all the technologies in the roadmap. The feature size we choose is 65 nm because

this represents the smallest feature size in the ITRS roadmap for both FeRAM and MRAM [14]. We will use the same methodology and models as explained earlier in this section to estimate the power consumption. To be consistent, we will assume that all properties of the technology are bound to the feature size, or technology node. To do this, we have to gather information from several different years in the roadmap. Therefore, the relevant parameters will be presented along with year they were listed in the roadmap.

In table 3.13, the resistance interconnect parameters are listed. These are now assumed to be equal for all the memory technologies, as all of the memory technologies are made with the same technology. It is still one parameters which can be different for the different technologies, and that it is the interconnect length. The reason for this is that the difference in area factor, X_{AF}, which gives different demands on the interconnects to be able to span the whole matrix.

Year in roadmap	Technology node F [nm]	Conductivity ρ_{eff} [µΩ cm]	Aspect ratio A/R	Source:
2007	65	2.2	1.7	[82]

Table 3.13: Interconnect resistance parameters for $F = 65\,\text{nm}$.

The model for estimating the read and write energy per cell for each of the memories are the same. The only difference now is that the input parameters technologies are assumed with a feature size of 65 nm. Following is a list of the memory technologies and the respective table where the parameters for $F = 65\,\text{nm}$ are located:

- DRAM parameters are listed in table 3.14.
- SRAM parameters are listed in table 3.15.
- FeRAM parameters are listed in table 3.16.
- MRAM and STT-MRAM parameters are listed in table 3.17.
- PCRAM parameters are listed in table 3.18.

The parameters for the access transistors, if the technologies are made with 65 nm technology, are listed in table 3.19.

	DRAM
Year	2007
X_{AF}	6
C_{cell} [fF]	25
V_{cell} [V]	0.65
V_{BL} [V]	0.65

Table 3.14: DRAM parameters for $F = 65\,\text{nm}$. From [113].

	SRAM
Year	2008
X_{AF}	140
V [V]	1.1
L_g [nm]	38
C_{gate} [fF µm^{-1}]	0.791
V_{BL} [V]	1.1
$I_{\text{sd-leak}}$ [pA µm^{-1}]	10

Table 3.15: SRAM parameters for $F = 65\,\text{nm}$. From [113].

	FeRAM
Year	2021
X_{AF}	15
σ [µC µm^{-2}]	0.175
A_{act} [µm^2]	0.33
V [V]	1

Table 3.16: FeRAM parameters for $F = 65\,\text{nm}$. From [14].

	MRAM	STT-MRAM	Sources
Year	2016	2012	[14]
X_{AF}	52	20	[14]
$E_{\text{cell-write/bit}}$ [pJ]	110	2.2	[14]
$R \cdot A_{\text{act}}$ [$\Omega\,\mu\text{m}^2$]	600	11	[14]
A_{act} [μm^2]	0.066	0.008	[14]
TMR [%]	90	120	[14]
R_P [kΩ]	9.090	1.385	Calculated
R_{AP} [kΩ]	17.27	3.025	Calculated
V_{read} [mV]	250	250	[85]
t_{read} [ns]	1	10	[86, 87]
$V_{BL,\text{write}}$ [V]	1.8	1.8	[9, 88]

Table 3.17: MRAM and STT-MRAM parameters for $F = 65\,\text{nm}$.

	PCRAM	Sources
Year	2009	[114]
X_{AF}	16	[114]
bits/cell	1	[114]
I_{reset} [μA]	202	[114]
t_{reset} [ns]	10	[114]
t_{set} [ns]	100	[9]
R_{reset} [kΩ]	300	[64]
R_{set} [kΩ]	6	[114]
V_{read} [V]	0.2	[91]
t_{read} [ns]	12	[93]

Table 3.18: PCRAM parameters for $F = 65\,\text{nm}$.

	Gate length L_g[nm]	Word line Voltage V_{WL} [V]	Gate capacitance C_g [fF μm^{-1}]	Source-drain resistance R_{sd} [$\Omega\,\mu\text{m}$]	Gate width W_g[nm]	Source:
DRAM	38	3	0.791	180	76	[105]
SRAM	38	1.1	0.791	180	114	[105]
FeRAM	32	0.8	0.789	190	64	[105]
MRAM	32	0.8	0.789	190	64	[105]
STT-MRAM	29	1.1	0.721	200	174	[105]
PCRAM	29	1.1	0.721	200	208	[105]

Table 3.19: Access transistor parameters for $F = 65\,\text{nm}$.

4 Methodology – Case Studies

In this section we will investigate several memory products which are on the market at the moment. This is done to get a picture of how the memories operate when connected to a microcontroller with the peripheral circuitry and other implications that follows. Ideally this should be done for embedded memories, but due to the large costs this would cause it was not an option. Therefore, standalone chips were chosen as a good compromise, also the theoretical models are done for standalone memories.

In many microcontroller applications, the microcontroller is inactive for long periods. Considerable amounts of energy can be saved if the memory can be turned off during these periods. This is possible for non-volatile memories, but if it is done for volatile memories the stored data is lost. The goal of these case studies was to measure the power consumption of the different memory chips when driven by a microcontroller (MCU) to read and write at different duty cycles. The duty cycle is defined as:

$$\text{Duty Cycle} = \frac{t_{\text{active}}}{T} \tag{4.1}$$

where t_{active} is the period the memory is active and T is the total period of the memory, and is given by the sum of the active and passive period.

In this work we have developed a set-up for experimental measurements of the memories. However, as a master thesis is only 20 weeks, there was not enough time to do the actual measurements. In the last part of this section we will present a study of what we could expect from the measurements, and try to give a picture of the memories, if they are operated as assumed in the theory section.

4.1 Measurement Set-up

The measurement set-up developed will be presented in this section. First an overview of the system will be presented, and then each of the components will be discussed in more detail.

4.1.1 System Overview

This section will give an overview of the measurement set-up in order to explain the details of later sections easier.

A block diagram of the measurement system is shown in figure 4.1. The RAM is to be controlled by a microcontroller-unit (MCU). There are three main groups

of signals between the MCU and the RAM, these signal groups are address, data and control. The address signals are sent from the microcontroller to the RAM to decide what address to read or write. The data signals contains the data which are either read or written by the microcontroller, and the control signals are sent from the microcontroller to the RAM; e.g. to tell the RAM to store the data currently on the data bus.

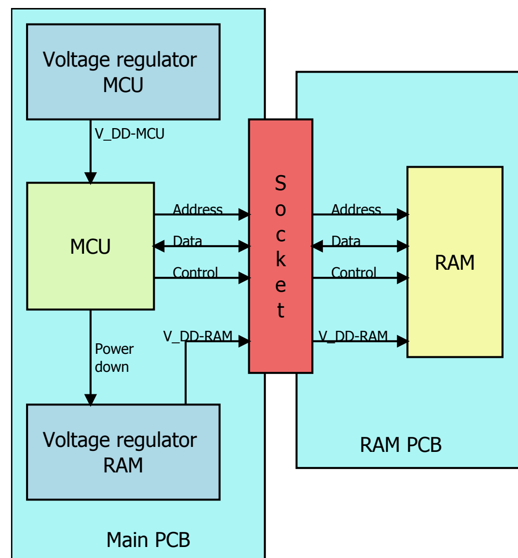


Figure 4.1: Block diagram of the measurement system. The microcontroller (MCU) and the RAM are placed on two different printed circuit boards (PCB) and connected together with a socket. The arrows indicate a signal or group of signals, and the arrows point in the signal directions.

The system has two different voltage regulators, one for the microcontroller maintaining a constant voltage, V_{DD-MCU}, and one for the RAM maintaining a constant voltage, V_{DD-RAM}. This makes it possible to measure the power consumption of each component independently, by probing the current and voltage supplied by each regulator. There is also a power down signal from the microcontroller to the RAM voltage regulator; this signal is for the non-volatile memories which can retain data without applied power. By using this signal it is possible to power down the RAM.

The system is created on two different printed circuit boards (PCB), which were made especially for this system. The reason for making custom circuit boards is because of the need to minimize the parasitic capacitance of the chip to chip interconnects. This is done by making as short interconnects as possible. How

much parasitic capacitance we can tolerate is estimated in a following section. The reason for choosing a solution with two different circuit boards connected with a socket is because this allows us to use the same microcontroller and voltage regulator for all the measurements of the different RAM chips. Using the same microcontroller and voltage regulator is done in order to make it so that any component variations in the measurement system will be as similar as possible for all the RAM chips.

4.1.2 Choice of RAM Chips

The choice of which RAM chips to survey was subject to multiple demands. The main demands were:

- Simple parallel interface. The memories utilize an interface to communicate with the microcontroller. It was decided that a parallel interface would be closest to the interface which would be used in an embedded solution. Furthermore, this interface should be as simple as possible in order to easier compare the different memory technologies.
- Similar memory capacity. The capacity of the memory impacts the energy consumption because of longer on chip interconnects and more peripheral circuitry are needed.
- Maximum operating frequency $f_{max} > 1\,\mathrm{MHz}$. As mentioned earlier the power consumption depends on the operating frequency. Ideally the memories should have identical maximum operating frequency. However, due to the large differences in memory technologies there are large variations in the maximum frequency. Therefore, to not exclude too many memory chips, a not too strict demand were set on the maximum operating frequency.

There were also some minor and practical demands, like the possibility to purchase the memory chips in small quantities. Still, not all these demands where possible to fulfil. One example is the STT-MRAM, which are only available to selected customers in 2013 [115]. Another example is the PCRAM where the only available chip had considerably larger memory capacity and a flash-memory like parallel interface. This interface uses a buffer when writing, and the write time varies from one write operation to another.

We further decided to choose two different chips from each of the emerging memory technologies, with different memory capacities and compare them with two different SRAMs. These memory capacities were 256 Kbit and 4Mbit. This was not done for the PCRAM, because it was only one possible option with a parallel

interface. No SRAM chips were available with 256 Kbit memory capacity; therefore, both SRAMs were chosen with 4 Mbit capacity. The selected memory chips are listed in table 4.1.

Label	Part number	Memory capacity	Maximum Frequency	Manufacturer	Data sheet reference
FeRAM A	MB85R-256F	256 KBit	6.7 MHz	Fujitsu	[116]
FeRAM B	FM22LD-16	4 Mbit	10 MHz	Ramitron	[117]
MRAM A	MR256A0-8B	256 Kbit	29 MHz	Everspin	[118]
MRAM B	MR2A16A	4 Mbit	29 MHz	Everspin	[119]
PCRAM	NP8P128A-13TSM60E	128 Mbit	8.7 MHz	Micron	[120]
SRAM A	IS61LV256-16AL	4 Mbit	100 MHz	ISSI	[121]
SRAM B	CY62146-EV30	4 Mbit	22.2 MHz	Cypress	[122]

Table 4.1: Selected memory chips and main attributes. The labels in the leftmost column represent the labels which are used throughout this work. The first word in the label is memory technology of the chip.

4.1.3 Choice of Microcontroller

The choice of microcontroller was not as strict. However, it was important that it had enough GPIO (General Purpose Input Output) ports to be able to support the parallel interface with multiple data and address lines. We also chose a microcontroller from our partner Energy Micro AS. When it was made sure that enough I/O ports were available, the choice went to a microcontroller with the most available on chip flash memory. This was done to avoid problems with the code size being too large. The selected microcontroller is listed in table 4.2.

4.1.4 Demands for the Voltage Regulators

The voltage regulators were to be designed at the instrumentation lab at NTNU, Department of Electronics and Telecommunications. The main job for this work

Label	Part number	Flash size	GPIO pins	Manu-facturer	Data sheet reference
MCU	EFM32GG980	1024 Kbit	83	Energy Micro	[123]

Table 4.2: Selected microcontroller (MCU) and summary of the main attributes.

was therefore to provide the demands which the voltage regulators needed to fulfil. It is the voltage regulators that supply the memories with power during the measurements, it is therefore essential that the voltage regulators supply only the amount of power that the memory demands.

The main demand for the voltage regulators are to supply a voltage within the supply voltage range for all current loads that will be connected to the regulator. The demand for the voltage regulators are therefore dictated by the range of the maximum and minimum current load. As the goal is to measure the power consumption when the memories are operated at different duty cycles, the voltage regulator for the memories must be able to supply a stable output voltage in the range from the standby currents up to the maximum active currents. These values are given in the data sheets of the memories and are listed in tables 4.3.

The voltage regulator of the microcontroller are subject to similar demands, however, as this work does not aim to measure the power consumption of the MCU, the microcontroller voltage regulator can have a minimum supply current. Extracted data from the microcontroller data sheet are listed in table 4.4.

	Supply voltage range [V]	Max current [mA]	Standby current [µA]	Max frequency [MHz]	Data sheet reference
FeRAM A	2.7–3.6	10	50	6.7	[116]
FeRAM B	2.7–3.6	12	270	10	[117]
MRAM A	3–3.6	65	6000	29	[118]
MRAM B	3–3.6	155	12000	29	[119]
PCRAM	2.7–3.6	50	160	8.7	[120]
SRAM A	3–3.6	100	1500	100	[121]
SRAM B	2.2–3.6	20	7	22.2	[122]

Table 4.3: Electrical characteristics of the selected memory chips.

The selected memory chips have different operating frequencies, and the maximum current, provided in the data sheets, are given that the memories are run on their

	Supply voltage range [V]	Max current [mA MHz^{-1}]	Typical current [mA MHz^{-1}]	Max frequency [MHz]	Data sheet reference
MCU	2.7–3.6	282	200	48	[123]

Table 4.4: Electrical characteristics of the selected microcontroller.

maximum frequency. It was decided that in order to get a comparable result from the measurements that the memories should operate at the same frequency. Still, it is possible that the memories requires up to the maximum current in power up phases etc. From table 4.3, we see that the voltage regulator should be able to keep a constant voltage of approximately 3.3 V if the load current varies from maximum 155 mA down to about 1 µA (The values in the table are for the maximum standby current, no minimum current are printed in the data sheets). The microcontroller maximum current depends on the frequency and how the microcontroller are operating, we will not operate it at the maximum frequency, so 5 A was assumed to be enough. The MCU regulator must also be stable at least down the sleep current, of 50 mA, of the microcontroller [123].

The voltage regulators will also have a time constant which decides how quickly they can adjust to changes in the power consumption. Because we can do the same operations over and over again in a loop, and then measure average power consumption, a time constant in the order of ∼milliseconds are acceptable. The time constant is mainly decided by the stabilizing circuitry of the voltage regulator and the decoupling capacitors used [124].

4.1.5 Demands on the Printed Circuit Boards Capacitances

The printed circuit boards (PCBs) were designed by the instrumentation lab at NTNU Department of Electronics and Telecommunications. The main job for this work was therefore to provide the demands which these PCBs needed to fulfil.

As mentioned, the reason that custom PCBs were needed in the first place was to make sure that parasitic capacitance of the chip to chip interconnects was kept low. This is important in order to assure that the measured currents would not be dominated by the charging of the parasitic capacitance in the interconnects from the RAM to the MCU.

We will here estimate a limit on the capacitance of the interconnects from the MCU to the RAM. We start by dividing the active currents in table 4.3 by the maximum operating frequency to get current per MHz. As mentioned earlier the

	Current per MHz [$\mathrm{mA\,MHz^{-1}}$]
FeRAM A	1.5
FeRAM B	1.3
MRAM A	2.3
MRAM B	5.4
PCRAM	5.6
SRAM A	1.0
SRAM B	0.9

Table 4.5: Active current per MHz for the selected memory chips.

active current is to the first order proportional to the operating frequency. The results are listed in table 4.5. The power associated with charging the interconnects between the chips are given by:

$$P_{\text{interconnects}} = \frac{1}{2} C_{tot} V^2 \cdot f \tag{4.2}$$

where C_{tot} denotes the total capacitance of the interconnects V is the I/O voltage level and f is the operating frequency. The capacitance of the interconnects should be small enough in order to clearly distinguish between the memories in table 4.5, the minimum difference here is $0.1\,\mathrm{mA\,MHz^{-1}}$. Taking this as a upper limit for the current per frequency and assuming $V = 3.3\,\mathrm{V}$, the upper limit for the total capacitance of the interconnects is given by:

$$C_{tot} < \frac{2}{V} \cdot \frac{I}{f} \approx 60\,\mathrm{pF} \tag{4.3}$$

The memories have to support for a word size of up to 16 lines, meaning that an upper limit for the capacitance is about $4\,\mathrm{pF}$ per line.

4.1.6 Microcontroller PCB Set-up

In this section we will discuss how the microcontroller is connected and operates. First of all, the microcontroller is connected to the power supply, debug interface and external clock sources, as discussed in the hardware design considerations application note [125]. One high and one low frequency crystal oscillator is connected to the microcontroller, as discussed in the oscillator design considerations application note [126].

Figure 4.2 shows a schematic which were developed as a part of this work. The schematics shows how the signals from the microcontroller should be connected on

the PCB designed by the instrumentation lab at NTNU. The signals to the RAM is routed through a socket, this makes it possible to use the same microcontroller for all the RAM chips.

As discussed earlier we have selected RAM chips with a parallel interface, we will therefore use the built in parallel interface on the EFM32 microcontroller called external bus interface (EBI). This interface will handle communication with the RAM chips like read/write enable, setup times etc. [127, 128]. As the different memory chips have slightly different number of I/Os and address lengths, we will use the EBI in a non-multiplexed N bit address 16 bit data mode. This means that we will use the $EBI_{AD}[15..0]$[1] lines to transmit data, and the $EBI_A[N-1..0]$ lines to transmit the address. The largest RAM chip is the PCRAM, which has a total capacity of 128 Mbit. This means that the PCB will need to support 23 address lines.

In addition to the address and data signals, the printed circuit board needs to support the control signals. The control signals from the EBI module which are used are write enable EBI_{WEn}, read enable EBI_{REn} and chip select $EBI_{CSn}[0]$. An extra four general purpose input output (GPIO) signals are routed to the RAM socket, so that they can be connect to any additional control input the memory chips may have. Examples of extra control signals include reset, write protect and byte select signals.

The microcontroller also have support for four LEDs and two push buttons as shown in figure 4.2, which can be used for debugging. These are connected to the microcontroller at unused GPIO ports. The final signal from the MCU is the power down signal, which is connected to the voltage regulator of the RAM and can remove the supply voltage from the RAM chip. This is needed to utilize the non-volatility of some of the RAM chips. The non-volatile memories will be in idle mode if the supply voltage of the non-volatile memories is not removed when the microcontroller is inactive. When the memories are in idle mode, they will have leakage currents similar to the SRAM, as shown in table 4.3. A schematic of which pins on the microcontroller which are used is attached in appendix B.

4.1.7 RAM PCB Set-up

The signals from the RAM socket are then connected to the RAM chip. Here, the data, address and control signals are connected to their respective pin on the chip. The ram chip also contains appropriate decoupling to avoid overshoots. A

[1] We will use the following notation for bus signals: Signal_Name[MSB..LSB], where MSB is the most significant bit, and LSB is the least significant bit.

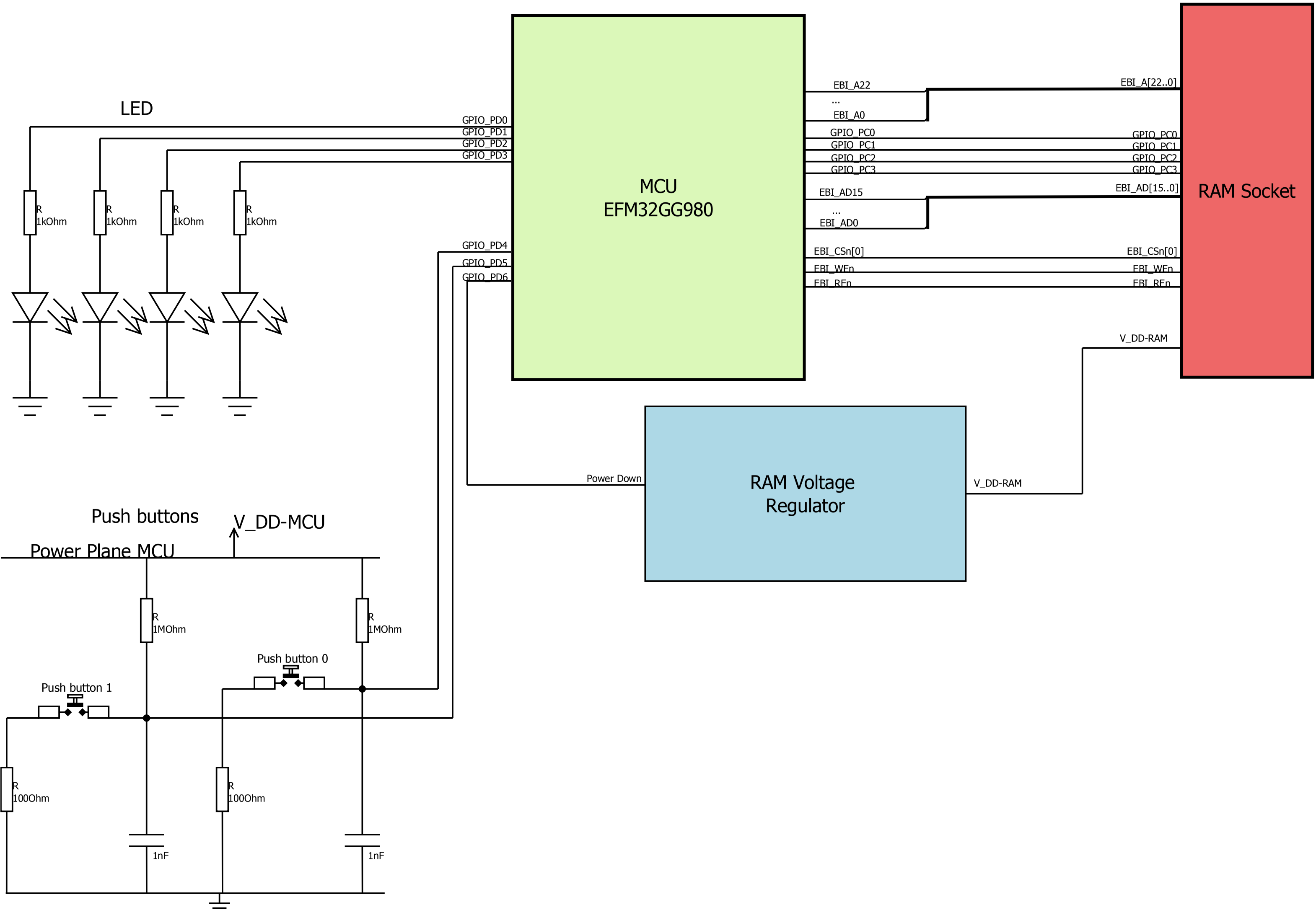


Figure 4.2: Schematic of the microcontroller (MCU) printed circuit board (PCB), the figure shows how the different signals from the MCU should be routed. Power supply and oscillators for the microcontroller and corresponding decoupling are not shown.

schematic of how this should be done for the PCRAM chip is shown in figure 4.3. The other RAM chips are connected in a similar fashion; all the schematics developed in this work are attached in appendix B.

4.2 Microcontroller Software

In this section the software developed to run on the microcontroller during the measurements is discussed. The goal was to measure the power consumption as the memory was operated at different duty cycles. The microcontroller is programmed in C.

Figure 4.4 shows an overview of the program flow of the software. As the microcontroller is started, the microcontroller is first initialized. Here, all the signals are declared, counters are set-up and data tables are generated. In the active state the microcontroller is programmed to continuously read and write the connected RAM for a given time period. In the passive state the microcontroller power downs the memory if it is non-volatile, after the memory is powered down the microcontroller enters sleep mode for a given time period. When the microcontroller is done sleeping it powers up the non-volatile memory again, and then returns to the active state after waiting the given power up time for the memory. If the memory is volatile, the microcontroller does not power down the memory, and hence does not need to wait for it to be powered up either.

The described cycle is repeated continuously, we will first write and read to the memory and return to the passive period over and over. This is because of the time constant of the power supply which makes it hard to measure the instantaneous power consumption. Still, the time constant does not affect the actual power consumption, as the power has to be delivered from the source [124]. We therefore cycle the memories and measure the average power consumption over a time period which is considerably longer than one cycle. We can then take the time average of the measured power consumption. This also allows for easier separating low currents from the noise.

In the following sections we will go more into detail about what is done in each of these states and the reasoning behind it. The full C code is attached in appendix C.

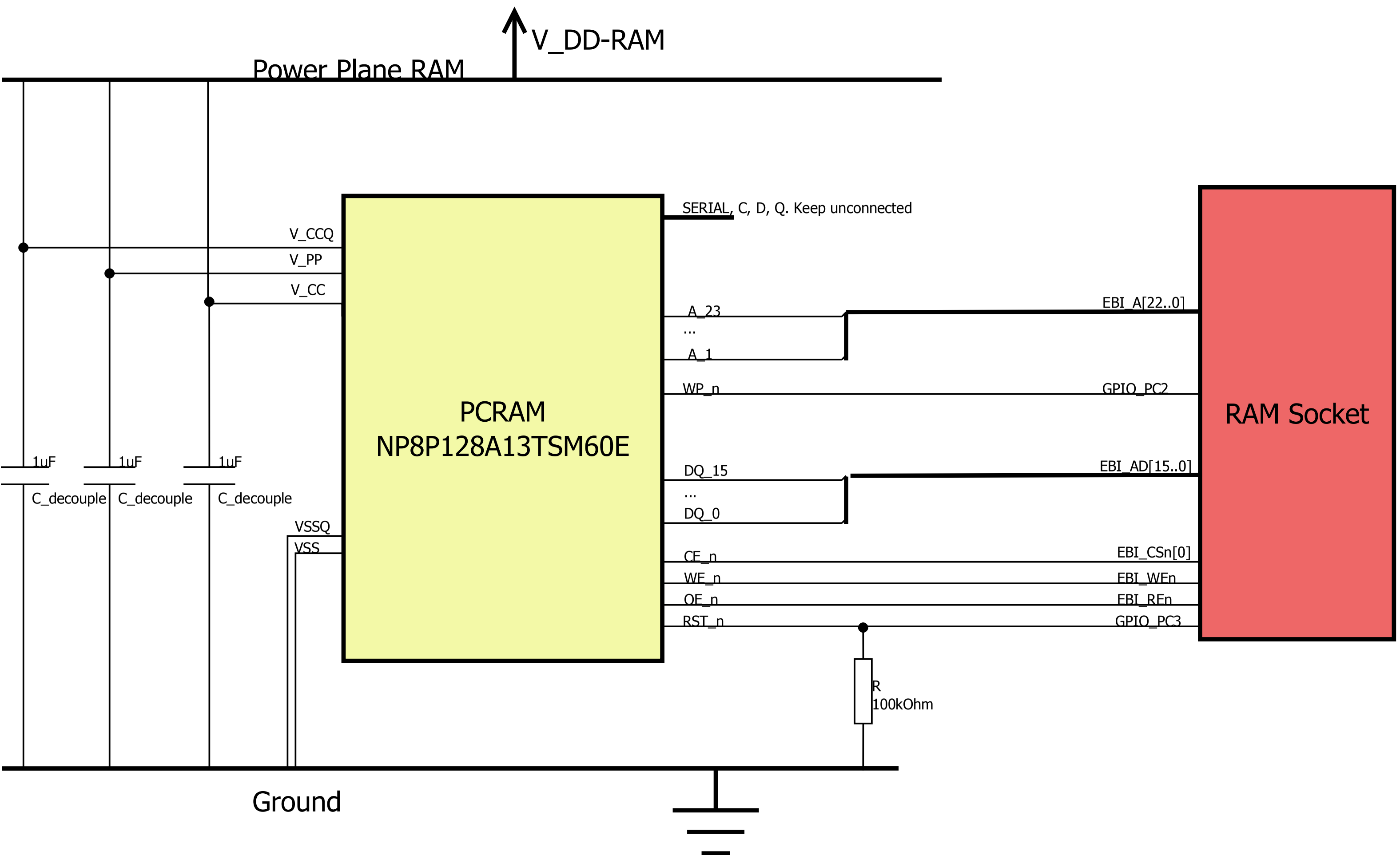


Figure 4.3: Schematic of how the PCRAM printed circuit board (PCB) signals should routed, and how the power supply should be decoupled.

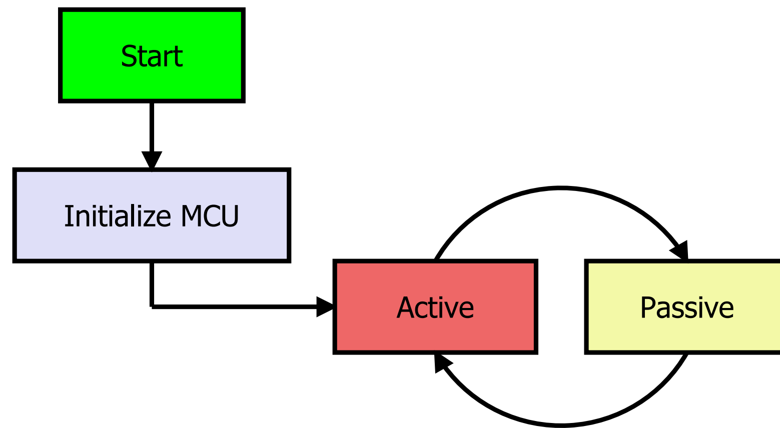


Figure 4.4: Program flow of the C code.

4.2.1 Initializing the Microcontroller

The first steps in the C code are to initialize the MCU. First, the core oscillator is changed from the default high frequency RC oscillator (HFRCO) to the high frequency crystal oscillator (HFXO). This is done because the crystal oscillators generally are more accurate than RC oscillators [126]. The HFXO is then divided down to the wanted core clock. This core clock will decide which frequency the memories are operated at, and should be low enough so that all the memories can be operated at the same frequency. This allows for easier comparison of the consumed power.

After the oscillators are correct, the TIMER0 of the microcontroller is set-up to overflow every millisecond, and to send an interrupt at each overflow. Each time an overflow occurs a counter counts up, this counter then decides what state the microcontroller is in. The TIMER0 overflow interrupt handler is shown here:

```
1 void TIMER0_IRQHandler(void)
2 {
3   // Clear flag for TIMER0 overflow interrupt
4   TIMER_IntClear (TIMER0, TIMER_IF_OF);
5
6   if(count<ACTIVE_COUNT){
7     active=true;
8   }else if(count<ACTIVE_COUNT+SLEEP_COUNT){
9     active=false;
10    Power_Memory_Down(true);
11  }else if(count<ACTIVE_COUNT+SLEEP_COUNT+WAKE_UP_COUNT){
```

```
12      active=false;
13      Power_Memory_Down(false);
14    }else{
15      // Reset counter
16      count=0;
17      active=true;
18    }
19    // Count number of overflows
20
21    count ++;
22
23  }
```

The code uses three constant integers ACTIVE_COUNT, SLEEP_COUNT and WAKE_UP_COUNT. As seen in the code the microcontroller uses the ACTIVE_COUNT to define how many milliseconds an active period should be, SLEEP_COUNT decides how many milliseconds the memory should be powered down and the how long microcontroller should be in sleep mode. WAKE_UP_COUNT decides how long the microcontroller must wait before the memory is ready to be active again, as the selected memories have a given time period after the power down signal is deasserted before they are available for reading or writing. A wanted duty cycle can then be achieved by appropriately selecting these parameters. Using the definition of the duty cycle from equation (4.1), the duty cycle can be calculated as:

$$\text{Duty Cycle} = \frac{\text{ACTIVE_COUNT}}{\text{ACTIVE_COUNT} + \text{SLEEP_COUNT} + \text{WAKE_UP_COUNT}} \tag{4.4}$$

Note that if the memory is volatile, the memory is not powered down, and the WAKE_UP_COUNT is set to zero. A consequence of this is set-up is that any of these time periods must be a multiple of milliseconds. Although, it is possible to program shorter active and passive time periods setting the counter to overflow more often than every millisecond. A couple of milliseconds is the order of the time constant that can be expected from the power supply, meaning that this limits how fast the memory can be turned on.

The next step in the initializing state is to set-up the EBI in the right mode so that communication with the RAM is achieved. The EBI is set-up in 16 bit data N bit address as discussed earlier. The set-up, strobe and hold times are all set to minimum, this can be done because the frequency of the writing and reading should be below the maximum frequency for each memory. This makes sure that all memories are operated under the same conditions. The prefetching is also disabled, to make sure that this does not interfere with the measurements.

4.2.2 Active Period

During the active period, the microcontroller is programmed to continuously write and read to the external RAM. The written data is generated during the initializing period of the microcontroller by the rand() function from the standard library, and placed in two arrays called test_A and test_B. The reason why two different arrays are used for writings is to make sure that the same data is not written to the same address each time. This can now be done by alternating by writing the data in array test_A and test_B. It is important to not write the same data to the same address each time because some memory technologies requires a different amount of energy whether a switching occurs or not. When alternating by writing data from array test_A and test_B, the cells will switch on average half of the time.

When using the external bus interface (EBI), the external RAM gets mapped to the memory space of the microcontroller. This means that to write a certain address of the external RAM, one writes to a corresponding address of the microcontroller memory [127,128]. The first address that writes to the external RAM is defined as EXT_RAM_BASE_ADDRESS. During the initializing period a set of random data is also placed in the array ram_address, by the same algorithm as placing data in the test_A and test_B.

The following code shows the write algorithm which is developed in this work:

```
1 void write_external_ram(void){
2   for (uint32_t j=0; j<PERCENTAGE_WRITE; j++){
3     if(!active){ // if the active period is over stop this
          function and return to main
4       return;
5     }
6     //Alternating by writing the A and B test array in order to
          make sure not to write the same data all the time.
7     if(table_A){
8       for  (uint32_t i=0 ; i< test_ARRAY_SIZE ; i++)
9       {
10
11        // Write data to the External RAM
12        *(uint16_t*)(EXT_RAM_BASE_ADDRESS + ram_address[i]) =
              test_A[i];
13
14      }// end for i
15    }else{
16       for (uint32_t i=0 ; i< test_ARRAY_SIZE ; i++)
17      {
```

```
18          // Write data to the External RAM
19          *(uint16_t*)(EXT_RAM_BASE_ADDRESS + ram_address[i]) =
               test_B[i];
20        }// end for i
21      } // end else
22      table_A = !table_A;
23    } // end for j
24  }
```

Writing to a random address is achieved by writing to the address (EXT_RAM-_BASE_ADDRESS+ram_address[i]). This is done to test the memory during random accesses. Sequential access could be achieved by writing to (EXT_RAM-_BASE_ADDRESS+i) instead. The writing process is repeated PERCENTAGE-_WRITE times, this is done to control the amount of writing compared to reading during the active period. The writing stops if the active flag is set low by the TIMER0 interrupt handler.

The developed read algorithm is similar to the write algorithm, but now it reads the external RAM instead of writing it, also the reading is repeated PERCENTAGE-_READ times instead of PERCENTAGE_WRITE times. The read algorithm is attached in appendix C along with the rest of the C code.

The software also have support for checking if the data written and later read back is the same. However, this is designed to only be done in the preliminary testing, when the microcontroller is connected for the first times. This is done to make sure that the communication with the memory is as expected, and should not be done during the actual power consumption tests, as this could influence the results.

The developed software will not be suited for the buffered write of the PCRAM, which has a flash-like interface. Here, write programs from the manufacturer are available [129]. This software still has to be integrated into the rest of the program to make sure that the duty cycle of the memory is maintained. However, due to the difference in the PCRAM configuration, the results will not be directly comparable with the results from the other memories.

4.3 Estimating the Power Consumption from the Data Sheets

We experienced some stability problems with the RAM voltage regulator which were designed by the instrumentation lab at NTNU department of Electronics and Telecommunications. The voltage regulator could not supply a constant voltage for the entire operating range we demanded [124], it showed a considerable ripple

on the output voltage, which could destroy the RAM chips. Due to these issues it was not enough time to do the required measurements before the deadline of this work. Therefore, we will instead use the data provided in the data sheets of the memory chips to estimate the power consumption, if the memories are utilized with a given duty cycle.

Because the characteristics listed in the data sheets are generally listed if the memory is operated at its maximum frequency, the data must be normalized first, so that the data for the given technologies can be compared. How this is done will be described in the following sections.

4.3.1 Write and Read Energy per Bit

The write and read energy per bit for the different memory chips can be estimated by taking the product active power consumption and random cycle time and then dividing by the number of I/Os [30], i.e.:

$$E_{\text{write/bit}} = \frac{P_{\text{active,write}} \cdot t_{\text{cycle,write}}}{N_{\text{I/O}}} \tag{4.5}$$

$$E_{\text{read/bit}} = \frac{P_{\text{active,read}} \cdot t_{\text{cycle,read}}}{N_{\text{I/O}}} \tag{4.6}$$

where $P_{\text{active,write}}$ and $P_{\text{active,read}}$ is the active write and read power respectively, $t_{\text{cycle,write}}$ and $t_{\text{cycle,read}}$ is the minimum write and read cycle time respectively and $N_{\text{I/O}}$ is the number of I/Os. This normalization makes it possible to compare the write energies of the different memory chips. Note that this is a different definition of the write and read energy per bit compared to $E_{\text{cell-write/bit}}$ and $E_{\text{cell-read/bit}}$ which is defined in section 3. In section 3, the cell write energy per bit is defined as the cost to write or read one cell, and is the energy when not taking the peripheral circuitry, charging of interconnects or switching of the access transistors into account. The write energy per bit we calculate in this section will take all these things into account and therefore these two write energies can not be directly compared.

To find the active power, the maximum current from the different data sheets are collected and multiplied with the maximum supply voltage. The typical values could in principle be used instead of the maximum values; however, not all data sheets have listed a typical for the supply current. Thus, the maximum value is selected to get comparable results from all the memory chips. If the data sheet does not differentiate between read and write current, the operating supply current is used in both cases. For the PCRAM, which has buffered write with variable

write times, the maximum buffer size is taken as number of I/Os and the typical buffer program time is taken as the write cycle time. The parameters and the resulting write power per bit is shown in tables 4.6 and 4.7.

	Supply voltage max [V]	Write current max [mA]	Write Cycle time [ns]	Number of I/Os $N_{\text{I/O}}$	Write energy $E_{\text{write/bit}}$ [pJ]	Data sheet reference
FeRAM A	3.6	10	150	8	675	[116]
FeRAM B	3.6	12	110	16	297	[117]
MRAM A	3.6	65	35	8	1023	[118]
MRAM B	3.6	155	35	16	1220	[119]
PCRAM	3.6	50	120000	512	35438	[120]
SRAM A	3.6	100	10	16	225	[121]
SRAM B	3.6	20	45	16	203	[122]

Table 4.6: Write energy per bit for selected memory chips.

	Supply voltage max [V]	Read current max [mA]	Read Cycle time [ns]	Number of I/Os $N_{\text{I/O}}$	Read energy $E_{\text{read/bit}}$ [pJ]	Data sheet reference
FeRAM A	3.6	10	150	8	675	[116]
FeRAM B	3.6	12	110	16	297	[117]
MRAM A	3.6	30	35	8	473	[118]
MRAM B	3.6	80	35	16	630	[119]
PCRAM	3.6	42	200	16	1890	[120]
SRAM A	3.6	100	10	16	225	[121]
SRAM B	3.6	20	45	16	203	[122]

Table 4.7: Read energy per bit for selected memory chips.

4.3.2 Estimating Active Power Consumption

With the write and read energy per bit calculated as in equations (4.5) and (4.6), the write and read power if the memory is operated with a frequency f and a word size of N_{ws} can be estimated:

$$P_{\text{write}} = N_{\text{ws}} \cdot E_{\text{write/bit}} \cdot f \tag{4.7}$$

$$P_{\text{read}} = N_{\text{ws}} \cdot E_{\text{read/bit}} \cdot f \tag{4.8}$$

since this is a theoretical estimate, we are no longer bound by the demand that all the memories should be able to operate at this frequency, we will therefore assume that same frequency and word size as in the theoretical estimates section, $f = 32\,\text{MHz}$ and $N_{\text{ws}} = 16$.

4.3.3 Estimating the Passive Power Consumption

The passive power consumption is different for the non-volatile and volatile memories. The volatile memories have a passive power consumption that depends on the standby current as they can not be powered down without losing data. In these estimates, to get an upper limit for this passive power consumption, the maximum values are chosen from the data sheets. The passive power consumption can then be estimated as:

$$P_{\text{passive,volatile}} = I_{\text{standby},max} \cdot V_{DD,max} \tag{4.9}$$

where $I_{\text{standby,max}}$ is the maximum standby current and $V_{\text{DD,max}}$ is the maximum supply voltage.

For the non-volatile memories the picture is more complicated. If the memories are powered down, they have a minimum power up time, $t_{\text{power-up}}$, which decides for how long the memory must wait after power is supplied before reading or writing operations can commence. The current consumed during this period is not defined in the data sheets. We will here assume that the current during this period is equal to the maximum current of the memory; this is done to get an upper bound for the energy consumption during power up. Assuming that the supply voltage rises instantly to $V_{DD,max}$ the power consumed in the passive period can then be estimated as:

$$P_{\text{passive,non-volatile}} = I_{DD,max} \cdot V_{DD,max} \cdot \frac{t_{\text{power-up}}}{t_{\text{passive}}} \tag{4.10}$$

where $I_{DD,max}$ is the maximum supply current and t_{passive} is the duration of the passive period. This also means that if the passive period is short compared to the power up time it will not be beneficial to power down the memory. In an energy optimized system for a non-volatile memory the standby currents are observed if the passive period is short. The passive power consumption for a non-volatile memory is in this case the minimum of equations (4.9) and (4.10). The corresponding parameters for estimating the passive power consumption are shown in table 4.8.

	Supply voltage max [V]	Max current [mA]	Standby current [µA]	Power up time [µs]	Data sheet reference
FeRAM A	3.6	10	50	50	[116]
FeRAM B	3.6	12	270	600	[117]
MRAM A	3.6	65	6000	2000	[118]
MRAM B	3.6	155	12000	2000	[119]
PCRAM	3.6	50	160	100	[120]
SRAM A	3.6	100	15000	NA	[121]
SRAM B	3.6	20	7	NA	[122]

Table 4.8: Parameters for estimating passive power consumption of the selected memory technologies, the power up times for the SRAMs are marked as not applicable, NA, as they are volatile.

5 Results and Discussion – Theoretical Estimates

In this section, all the results from the Methodology – Theoretical Estimates, section 3, are presented. The results are either tabulated or presented in a graph. First, the results from 2012 and 2017 are presented, where 2012 represents what is available today and 2017 represents what is available in five years. 2017 is chosen because five years is a typical time it takes for a semiconductor company to implement a new technology [2]. When the results are presented, each of the subcomponents in the power estimates given by equations, (3.14) and (3.15), are evaluated and discussed before the total is summed up and presented. All the following results are under the assumption that the memory block is 32 kB = 256 Kbit, and are operated at a write frequency of $f = 32\,\text{MHz}$ with a word size of 16 bits.

After the results from 2012 and 2017 have been discussed, the results from section 3.12, where the memory technologies are assumed to be manufactured with the same feature size, are presented. This is done to compare the memories if they are made with the same processing technology.

If the results are a subject to any other assumptions than the ones presented in the methodology section, these assumptions are then mentioned. After each result is presented, it is discussed with an emphasis on the validity of the models. If there are any effects not covered by the model, which could impact the results, these are mentioned.

When discussing the results the emphasis is put on the relative magnitudes of the results, rather than the actual values. Because of all the assumptions made in our model the actual values will have reduced validity. But as most of these assumptions are equal for all the memory technologies, the relative differences are assumed to be similar, maybe shifted with some constant active power. In the last part of this section some of the most important challenges for each memory technology are mentioned. These describe what needs to be achieved by the technology in order for the estimates to be valid.

5.1 Area Consumption

The density of the memory is of utmost importance when estimating the power consumption, this is because one of the main contributors to the power are the capacitance of the interconnects. The density of the memory is also one of the most important factors for a memory technology, as it is directly proportional to the price per bit [13].

The results from the area consumption from table 3.1 and 3.2, are displayed in figure 5.1. The graph shows the area consumption in terms of bits per µm^2 for the different memory technologies. The estimates from 2012 are displayed in blue, while the estimates from 2017 are displayed in red.

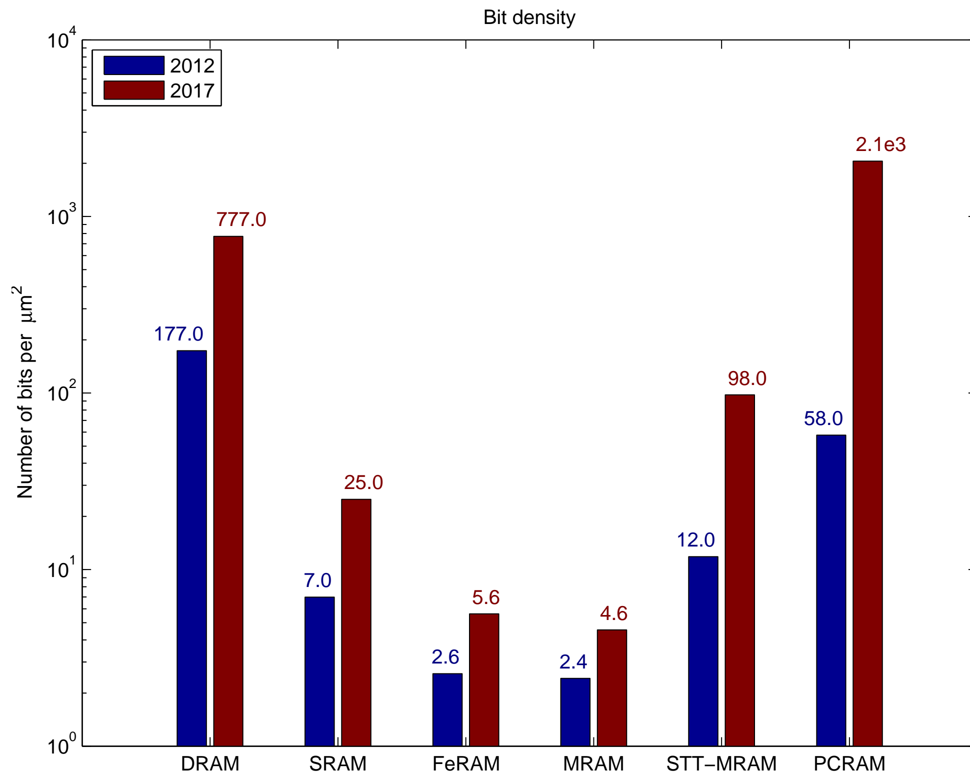


Figure 5.1: The area consumption of the different memory technologies in terms of bits per µm^2, shown on a logarithmic scale. The numbers above each bar represent the height of the corresponding bar.

All the different memory technologies increase their density as they are scaled down, as shown in figure 5.1. This result is as expected. The technology with the largest increase in area is PCRAM. The reason for the leap in density is that it should be possible to store four bits per cell in 2017, as shown in table 3.2. This is combined with the fact that PCRAM is at the same feature size as DRAM. PCRAM is the only non-volatile memory expected to surpass DRAM in terms of bit density.

FeRAM and MRAM are the memory technologies with the lowest bit densities ($\approx$ 2.6 and $\approx$ 2.4 bits/µm^2 in 2012, and $\approx$ 5.6 and $\approx$ 4.6 bits/µm^2 in 2017 for

FeRAM and MRAM respectively). The reason for this is that they are expected to be manufactured with a considerably larger lithographic feature compared to the other memory technologies discussed. The MRAM also have a relatively large area factor ($X_{AF} \sim 50$), since it need an extra digit line to contact the cell for writing. These low densities are a factor of ~ 100 lower than DRAM, but it is still comparable with SRAM. The FeRAM and MRAM have a density within a factor of ~ 5 compared with SRAM.

The STT-MRAM(≈ 12 bits/µm^2 in 2012 and ≈ 98 in bits/µm^2 2017) is denser than the SRAM(≈ 7 bits/µm^2 in 2012 and ≈ 25 in bits/µm^2 2017). This is despite that the SRAM has a lower feature size than the STT-MRAM; however, the area factor of SRAM is larger than the area factor of the STT-MRAM.

The area calculations only take the area of the storage elements and access devices into account. It does not look at: Decoders, sense amplifiers, I/O control or other peripheral circuitry, which are needed for a functional memory [15]. However, many of these aspects would be needed for all the technologies, and will scale with the feature size [13]. There are some aspects which can contribute to other area differences, which are not taken into account in this model:

- Sense amplifiers. The sense amplifiers can be different for each technology. A larger and more complex sense amplifier is needed if the signal difference between a logic one and a logic zero is small [13].
- Multi-level storage. The PCRAM is expected to use multi-level storage in 2017, this can affect the area for many of the components in the memory. Examples include sense amplifiers which now need to be able to separate between sixteen levels instead of two, decoders which would need to address the same line for multiple addresses and control logic which is needed for the write and verify algorithm.
- Write-back control. For the DRAM and FeRAM which have a destructive read-out, a system which makes sure that the cells are written back after each read is needed. This can be incorporated in the sense amplifier [13].
- Refresh control. The DRAM needs some kind of control logic in order to make sure that all cells are refreshed in time [15].

5.2 Capacitance and Resistance of the Interconnects

The capacitance and resistance of the interconnects are listed in table 5.1. Because it is assumed that the matrix and cells are square, the capacitances of the bit lines and word lines are equal. The general trend observed in table 5.1 is that the

resistance and capacitance is coupled to the bit densities. Increased bit density gives larger resistance and lower capacitance, this is as expected.

As seen in table 5.1, the capacitance values calculated with our model are in the 100 fF range. The resistance of the interconnects calculated is with equation (3.17). The resistance is proportional to the length L, but also inversely proportional to the feature size squared. This means that the lowest resistance is achieved by increasing the density through other means than the feature size. This gives the largest resistance values for the SRAM, where the density is mainly achieved by aggressively scaling the feature size, and the lowest resistance value for the PCRAM where a large density is achieved by multi-level storage.

The capacitance is also dependent on the density of the memories, as seen in equation (3.19). Longer interconnects are needed when the density is low, the largest capacitances are estimated for the SRAM, FeRAM and MRAM, which have the lowest densities.

	Resistance [Ω]		Capacitance [fF]	
	2012	**2017**	**2012**	**2017**
DRAM	495	625	85	50
SRAM	2190	3940	425	290
FeRAM	390	550	400	310
MRAM	790	735	475	395
STT-MRAM	455	650	245	115
PCRAM	604	385	140	30

Table 5.1: Capacitance and resistance of interconnects results.

ITRS reports that for $F = 180\,\text{nm}$ and $X_{\text{AF}} = 8$, a bit line capacitance of 320 fF [73] is expected. Using equation (3.19) to model the same capacitance gives a value of 285 fF, this is a deviation of 11 %.

The deviation can be explained by an increased fringing capacitance, as this capacitance can be as large as 25 % - 50 % of the total capacitance [77]. Another point that can explain the deviation are the assumptions of a square matrix and cell. A real matrix does not have to be square [13], meaning that either the bit line or the word line can be longer than the other.

The resistance of the interconnects is calculated using the same model as Zhirnov et al. as well [38], so it is assumed that the deviations in resistance are of the same order. The resistances are not a subject to fringing fields like the capacitors, but deviations in the resistance can occur because of contact resistances etc.

Other parasitic capacitances and resistances than the ones from the interconnects

are neglected. This is because the other effects are considered minor compared to the effects of the interconnects [13]. However, contact resistances and capacitances can become a significant if the memories are embedded in the upper metal layers, as shown in figure 2.30 and 2.37. This will increase the total power consumption.

5.3 Cell Write Energy per Bit

The estimated cell write energies per bit for the different memory technologies are shown in figure 5.2. The cell write energy per bit is here defined as the energy needed to switch a cell per bit.

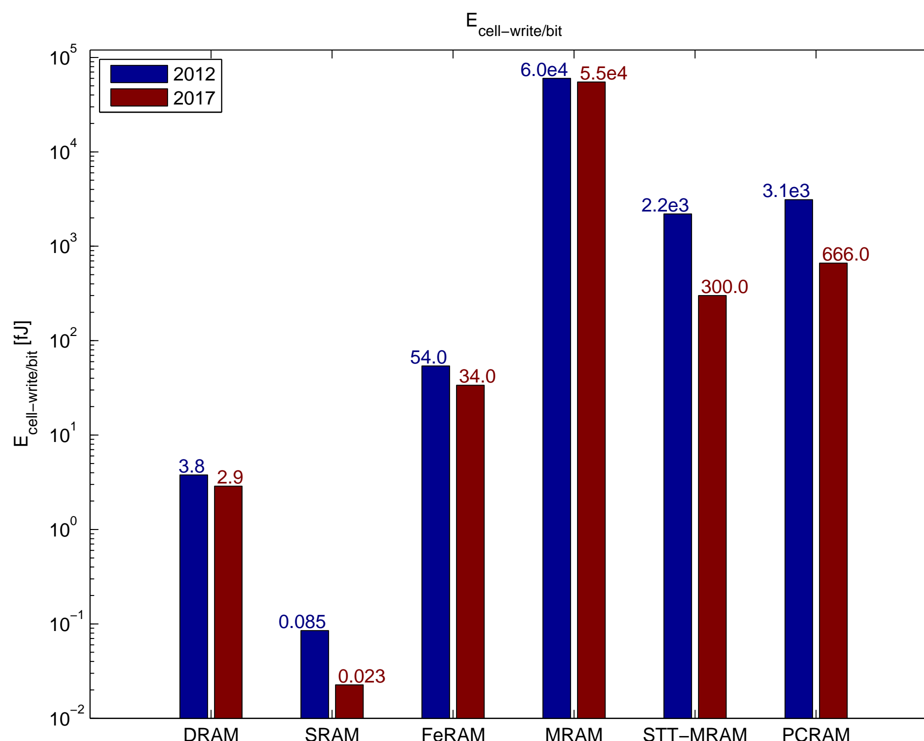


Figure 5.2: The write energy per bit of the different memory technologies, shown on a logarithmic scale. The numbers above each bar represent the height of the corresponding bar.

As seen in figure 5.2, the conventional memory SRAM is expected to have the lowest cell write energy per bit. With this model it is about one to two orders of magnitude lower than DRAM, which has the second lowest write energy. The

reason for this low cell write energy, for the SRAM, is that it does not cost much energy to switch the gate capacitance of a few transistors. Of the emerging memories, FeRAM consumes the least energy per bit followed by STT-MRAM, PCRAM and finally MRAM. This order is the same in 2012 and 2017.

The memories have different relative reduction from 2012 and 2017. The largest reductions are for STT-MRAM and PCRAM which are in the order of ~ 5, the reasons for this that is the write energies are substantially reduced when scaling down the storage element. MRAM and DRAM have the lowest relative reduction from 2012 to 2017, which is in the order of 1. For DRAM, the reason is that the cell capacitance should remain constant while the cell is scaled down. For MRAM the write current increase as the cell is scaled down, leading to poor scaling properties.

5.3.1 DRAM

The DRAM cell write energy per bit is estimated by equation (3.20). The assumptions made are: That the energy needed to write the DRAM cells are equal to the energy cost of charging the DRAM capacitor from 0 V to $\pm V_{\text{cell}}$. This is rarely true, as there is generally some stored charge left on the capacitor. This means that the energy cost is overestimated. The largest possible overestimation is when the capacitor already is at V_{cell}, then there will be no energy cost because no charging is needed.

Our model assumes that the capacitor value is 25 fF in both 2012 and 2017, as indicated by the ITRS [9]. However, as the capacitance of the bit lines are reduced with scaling, a value of 20 fF is considered to be enough [13]. Lowering the capacitance to 20 fF will lower the energy to write a bit by 20 %.

Other effects such as a non-ideal capacitors and variations in the capacitor value are also neglected. These effects are considered minor and are neglected because we are interested in the order of magnitude for the write energy.

5.3.2 SRAM

The SRAM cell write energy per bit is estimated as charging the gate capacitance. The gate capacitance is relatively low compared to bit line and word line capacitances and is energy inexpensive to charge [14]. This is the reason why the write energies for the SRAM is so low.

The SRAM transistors are here assumed to be made with low standby power technology. This will increase the cell write energy per bit by approximately 67 %, compared to if the transistor would be made with low operating technology [14], as seen in table 2.4 on page 29. The low standby power transistors will also have a limit on the maximum clock frequency compared to high performance or low operating power transistors.

As mentioned in the theory section, the gate width of the transistors which makes up the SRAM, has a large impact on the SRAM yield. It is here assumed that the gate widths are three times the gate length. With optimization of silicon on insulator or multi-gate transistors, it is possible to reduce the gate width to two times the gate length and have the same yield [25]. This is possible because these technologies can control short channel effects without doping the channel. Reduced channel doping reduces threshold voltage variations [26], as shown in figure 2.6 on page 18. This reduction in gate width will reduce the gate capacitance and hence also the cell write energy by 33 %.

5.3.3 FeRAM

The cell write energy per bit for the FeRAM is estimated by equation 3.26. Like the DRAM, the FeRAM write energy per bit is estimated as charging a capacitor. The FeRAM model does not take into account the difference in energy of a non-switching event and a switching event, as described in theory section. This has the same effect as for DRAM, that we are overestimating the write energy per bit. The parameter given for the switching charge density by the ITRS, is the assumed lowest switching charge that allows for a detectable signal [73]. The switching charge depends on the ferroelectric material, and increasing the switching charge will in turn increase the cell energy.

The FeRAM requires pulsing of an extra interconnect, the plate line, in order to read and write. The energy cost for charging the interconnects are not taken into account in the cell energy consumption, but it is a part of the total power model.

5.3.4 MRAM and STT-MRAM

The cell write energy per bit for the MRAM and STT-MRAM is taken directly from the ITRS [14]. The ITRS estimates the write energy per bit for the MRAM and STT-MRAM as: $I_{\text{write}} \cdot V_{\text{write}} \cdot t_{\text{write}}$ [14].

For the conventional MRAM we have assumed that a toggle algorithm is used to switch the cell. For toggle MRAMs, the cell has to be read each time before it is written. It does cost some energy to read the cell, but the energies needed to create the magnetic fields in order to switch the cell are several orders of magnitude larger. It is therefore energetically favourable to read the cell before writing it, as this reduces the cell write energy by 50 %. The read before write operation does however limit the maximum operating frequency of the memory, as it will take some time to read the cell.

5.3.5 PCRAM

The cell PCRAM write energy has two components, because different energies are required to reset or set the phase change material. The reset current is collected from the ITRS roadmap [14], while the set current is estimated with equation (3.32). From this equation we see that the set current is proportional to the reset current. This result was also stated by Wong et al. [66]. The proportionality constant where assumed to be $1/\sqrt{3} \approx 0.58$. This proportionality constant varies some with the cell design, but it is well within the limits of the data presented by Lee et al. [65]. If the set current proportionality constant is taken to the extremes of the data presented by Lee et al., the set energy can be reduced by 67 % or increased by 33 %.

It is assumed that the resistance when setting or resetting the bit is equal to the set resistance [14]. However, as seen in the IV-curve in figure 3.2 on page 83, this is an approximation. The resistance will be lower at higher currents due to the increase in temperature. This reduction in resistance, at higher temperature, will slightly reduce the energy needed to switch the cell.

The PCRAM is expected to use multi-level storage in 2017, storing four bits in each cell [10]. A special methodology was developed in this work to estimate the extra write energy consumption associated with multi-level storage. As to the author's best knowledge, no commercial four bits per cell has yet been produced, only proof of concept devices, like the work done by Nirschl et al. [68] has been demonstrated. Therefore, multiple assumptions are made, mainly based on published results with 2 bit per cell [67, 69, 70, 94–100]. The major assumption done here is the choice of write and verify algorithm, as this decides the type and number of pulses used to write the cell. A stair-case up(SCU) algorithm where assumed, not only because it is practical with same pulse length for all the operations, but also because it was one of the best documented algorithms [69, 96, 97, 100].

The other assumptions made when developing a model for MLC-PCRAM, where

specific for the SCU algorithm. The assumption that on average 15 iterations are needed, have the largest impact on the energy consumption as this determines how much energy is applied to the PCRAM cell. This assumption was motivated by the fact that the average number of operations increase exponentially with the number of bits per cell [101], and that state of the art two bit per cell reported an average of 2-3 iterations [69].

There are two reasons that the energy consumption still decreases even though an average of 15 iterations are assumed: The first reason is that less energy is applied in each pulse compared to single levels cells. The other reason is that four bits are written at a time, thus reducing the energy cost per bit. However, the write and verify algorithm will have a large impact on the total time it takes to write the cell [101]. This problem can be reduced by clever use of buffers, but will be a considerable problem for implementing multi-level phase change technology as the working memory for the processor [67, 69, 101].

5.4 Cell Read Energy per Bit

In figure 5.3 the read energy per bit for the different technologies are shown. The energies are shown in blue for 2012, and red for 2017.

SRAM has a cell read energy of zero, this is because no switching of the cell transistors occurs. This is not equivalent with the statement that SRAM does not require energy to read, but the energy which is consumed when the SRAM is read is because of charging and discharging of the bit lines. DRAM and FeRAM does also not require any explicit energy to read the cell, as they also discharge the bit lines. The discharging is a spontaneous process when the bit lines are charged and the access transistor channel made conducting. The charging of the bit lines are taken into account by another part of the model. However, both for DRAM and FeRAM the read-out is destructive, meaning a write-back is needed after each write operation. This is why the write energy per bit is listed as a read energy for the DRAM and FeRAM technologies.

For the resistive memories, MRAM, STT-MRAM and PCRAM, a reading energy is associated with a voltage which are applied to a resistor instead of discharging of capacitances. This reading energy is lowest for the MRAM and then PCRAM, the largest of the resistive memories is STT-MRAM. The differences are mainly due to different resistance values estimated for the different technologies.

A sequential read algorithm is assumed for the PCRAM in 2017, this sequential algorithm requires four reads to determine the resistance value [69]. However, as four bits are read at a time these two effects cancel, as seen in equation (3.41).

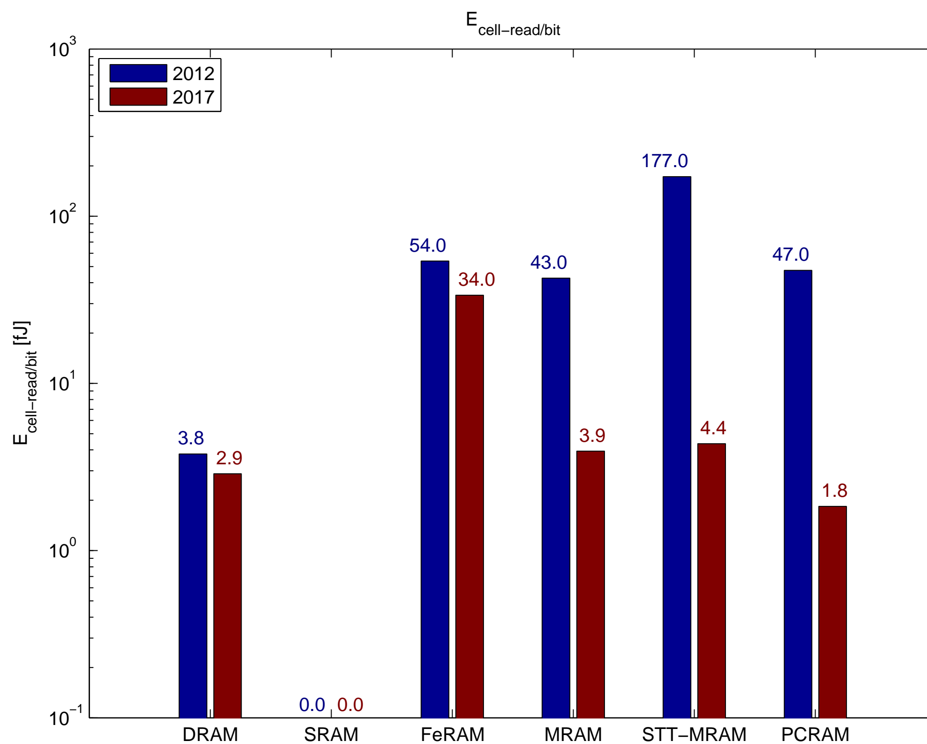


Figure 5.3: The read energy per bit of the different memory technologies, shown on a logarithmic scale. The numbers above each bar represent the height of the corresponding bar. The write energy is listed for the DRAM and FeRAM, because they need a write-back after each read operation.

If a parallel sensing algorithm is used to reduce the read time, fifteen reads are necessary to determine which of the sixteen resistance values was stored [69]. This will reduce the read time, but it will increase the read energy by a factor of 3.75 and increase the area required for the sense amplifier by a factor of 15. Therefore, the parallel regime is not considered a viable solution [69].

5.5 Energy Cost to Switch the Access Transistors

The energy cost to switch the gate capacitance of the access transistors are shown in figure 5.4. This is the energy cost to switch all the access transistors connected to one word line, as stated in equations (3.12) and (3.13). The figure shows the results for 2012 in blue and the results for 2017 in red.

The access transistors are assumed to be similar to the transistors used in logic

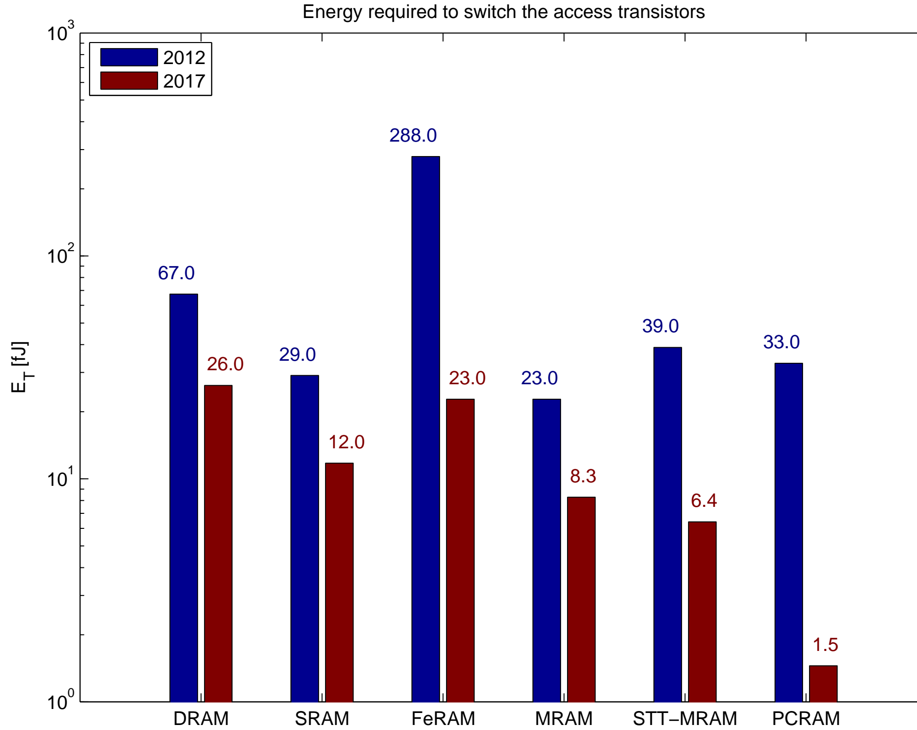


Figure 5.4: The energy required to switch the gate capacitance of the access transistors on a logarithmic scale. The numbers above each bar represent the height of the corresponding bar.

circuits. This might not be the case, because transistors, especially designed as access devices, could be implemented in all of the technologies. This could in turn affect the energy required to switch them, as they could be designed to have another gate capacitance or require another voltage applied to the word line. An example of the latter is the DRAM transistor which uses highly boosted voltages to deliver enough current while also maintaining a low leakage. The highly boosted voltages of the DRAM is accounted for in this model, by taking the DRAM word line voltage from the DRAM tables in the ITRS [14]. However, it could be possible that not only the DRAM word line voltage is altered, but also the gate capacitance. It is assumed that the gate capacitance for a VCAT will be of the same order of magnitude as the multi-gate transistors designed for logic circuits, as they are both are multi-gate transistors with similar dimensions.

In figure 5.4, we see that the energy required to switch the access transistors scales with the feature size. This is because scaling down the cell allows for a smaller gate length and gate width. The highest transistor energy is therefore observed

for the FeRAM, which has the largest feature size. The lowest switching energy is observed for the PCRAM in 2017. The reason for this is that the four bit per cell decreases the number of transistors needed by a factor of two. The energy shown for the PCRAM is the energy required to switch the access transistors once, similar to what is done when reading. However, when a write and verify operation are done for the multi-level PCRAM this energy has to be supplied for every read and write step of the write and verify algorithm.

DRAM has a relative high transistor switching energy; the reason for this is the mentioned highly boosted voltages which is needed. The SRAM also have a short feature size, but has high access transistor energy. The reasons for this is that the SRAM have two access transistors per cell, and that the SRAM needs wider access transistors ($W_g = 3L_g$) to ensure a high yield. If a gate width of $2L_g$ was utilized instead, the resulting gate energy will decrease by 33 %. Similar for the DRAM, FeRAM and MRAM, we assumed that the gate width was equal to $2L_g$. If this width is increased to $3L_g$ the energy required for switching the capacitance will increase by 50 %.

For the STT-MRAM and PCRAM, we calculated the transistor width so that the transistors could deliver enough current to switch the cells. It was then assumed that high performance transistors where used, as these transistors have the highest on current [14]. However, as these switching currents are expected to be reduced when scaling down the cell, the STT-MRAM and PCRAM transistors does not require as wide transistors in 2017. This explains the large reduction in the energy cost for STT-MRAM and PCRAM to switch the access transistors from the year 2012 to 2017.

5.6 Write Power Consumption

The estimated write power for the different memory technologies are shown in figure 5.5. This is the power consumed, according to equation (3.14), which represents the sum of the power required to:

- Charge the capacitance of the word lines.
- Charge the capacitance of the bit lines.
- Switch the cells.
- Switch the access capacitors.

Looking at the estimated write power in figure 5.5, we see that the conventional memory, DRAM, is the one with the lowest write power consumption both in

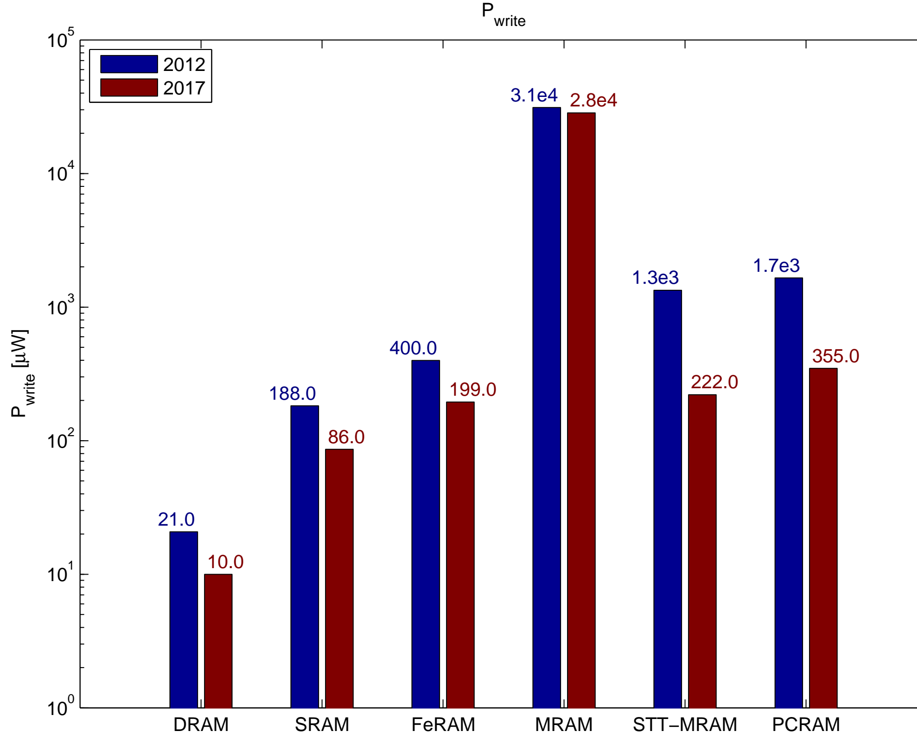


Figure 5.5: The write power of the different memory technologies, shown on a logarithmic scale. The numbers above each bar represent the height of the corresponding bar.

2012 and 2017. SRAM has a write power of one order of magnitude higher than DRAM, but it is still lower than all of the emerging non-volatile memories. FeRAM has the lowest write power consumption, followed by PCRAM and then STT-MRAM. However, the difference between FeRAM, STT-MRAM and PCRAM are drastically reduced from 2012 to 2017. The highest write power is estimated for MRAM, the write power for MRAM is one to two orders of magnitude larger than STT-MRAM and PCRAM.

The reasons for the different write powers can be explained by looking at figure 5.6. Here, the different terms in equation (3.14) is plotted on a logarithmic scale. The write power terms for 2012 are in figure 5.6a, while the terms for 2017 are plotted in figure 5.6b. In the following sections, we will discuss the breakdown of the write power for all the different memory technologies.

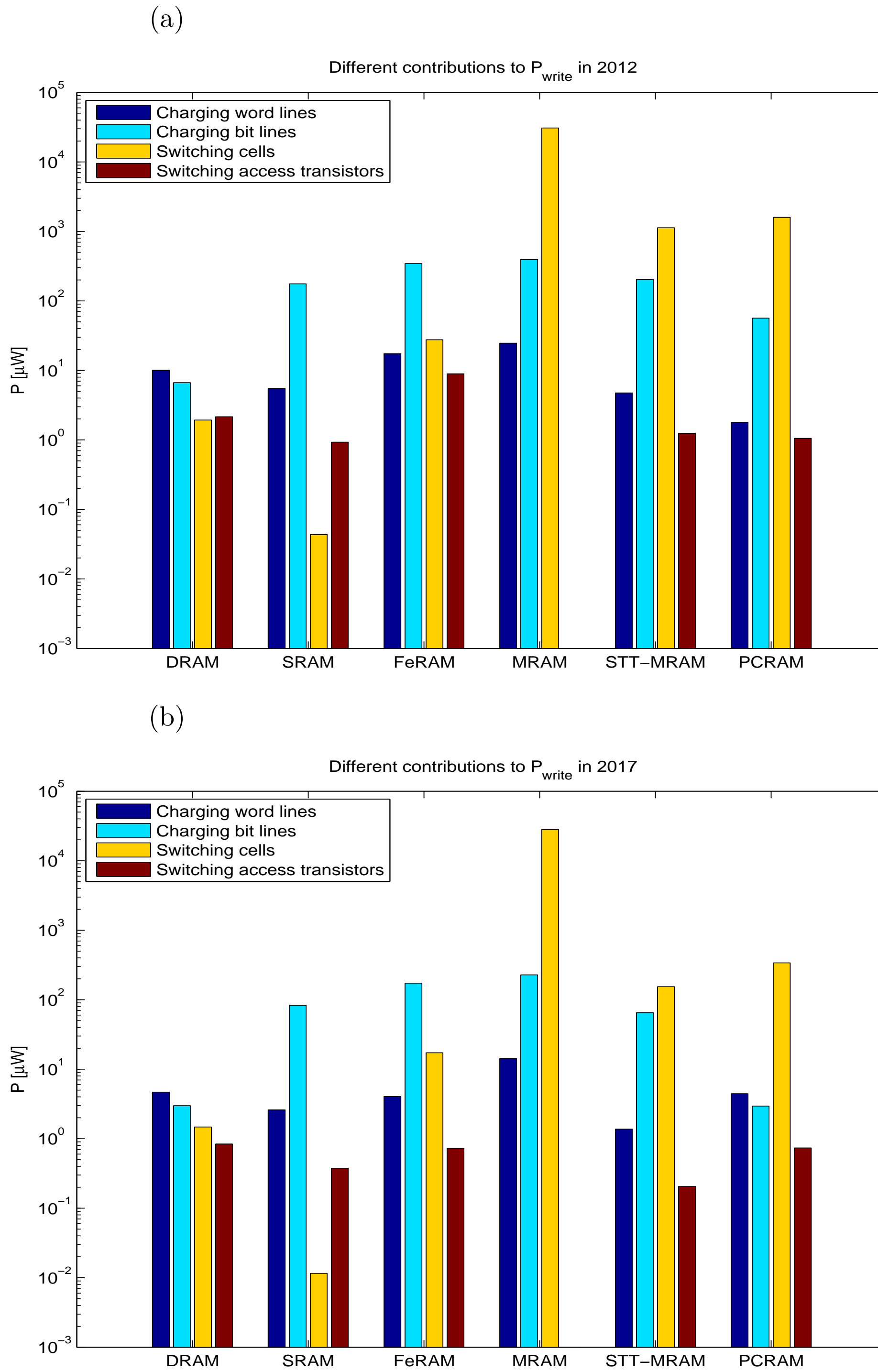


Figure 5.6: Breakdown of write power consumption shown on a logarithmic scale. (a) Shows the results for 2012. (b) Shows the results for 2017.

5.6.1 DRAM

For DRAM, we see in figure 5.6 that the dominating factor in the write power is the word lines, followed by the bit lines. The power to required to switch the cell and transistors are each about a factor of 5 lower than the power consumed by charging of the interconnects in 2012. However, this factor is slightly reduced in 2017.

The DRAM is one of the few memories that have a lower power consumption for charging the bit line compared to charging the word line. So the power cost to charge the bit line is generally higher than the cost to charge the word line. The reason for this can be explained by the power required to charge the word line, given by equation (3.6), compared to the power required to charge the bit line, given by equation (3.8). In these equations we see that the power required to charge the bit line is scaled by the number of bits written at a time, this number is known as the word size, N_{ws}. When we have assumed a square matrix the bit line capacitance and word line capacitance are equal. This means that if the voltage applied to the bit line and word line are equal; the power consumed to charge the bit lines will be N_{ws} times larger than to charge the word line. However, for the DRAM, the voltage applied to the word line is considerably larger, because of the highly boosted voltages needed for the DRAM access transistors [14]. This again causes DRAM to have a higher power consumption for charging the word lines compared to the charging of the bit lines.

Still, DRAM is the only memory technology where the energy consumption is not clearly dominated by one or two terms. This means that all the contributions must be taken into account when evaluating the DRAM power.

5.6.2 SRAM

For SRAM, we see in figure 5.6 that the dominating factor is the charging of the bit lines. The reason for this is the relative high interconnect capacitance, as seen in table 5.1, as well as the cell structure of the SRAM which requires two bit lines. For SRAM, the charging of the bit lines is over one order of magnitude larger than the other components. It is this factor that makes the total SRAM write energy, as shown in figure 5.5, of the same order of magnitude as the non-volatile memories.

We also see that for the SRAM, the switching of the few transistors in each cell does not contribute significantly to the total power consumption. The power cost to switch the cells is about three orders of magnitude lower than the power cost

to switch the bit lines.

5.6.3 FeRAM

For FeRAM, we see in figure 5.6 that the picture is similar to that of SRAM. It is the charging of the bit lines which is the dominating factor. The FeRAM uses a third line, called the plate line, to switch each cell. This energy is added to the bit line energy to show the picture for the FeRAM on the same format as the other memories. Regardless, this effectively works as if the FeRAM has two bit lines. The FeRAM is also the memory technology which is made with the largest feature size, which results in relatively large parasitic capacitance of the interconnects, as seen in table 5.1.

The difference from the SRAM to the FeRAM is that the power cost associated with switching of the cell for the FeRAM is about than one order of magnitude lower than the power cost of charging the bit lines of the FeRAM. This means that the switching of the cell has some impact on the total power consumption, which is plotted in figure 5.5.

5.6.4 MRAM

For MRAM, we see in figure 5.6 that the dominating factor is the energy required to switch the cells. It is two orders of magnitude larger than any other component. This component is almost constant when the cell is scaled down, and is the reason for the high power consumption observed for MRAM, in figure 5.5, as seen in both 2012 and 2017.

As the MRAM does not use any access transistors while writing, this power cost is zero. However, as the energy to switch the cell is so large, this has little impact on the total power consumption.

5.6.5 STT-MRAM

For STT-MRAM we see in figure 5.6 that as the energy required to switch the cells is considerably reduced compared to field switching MRAM. This also reduced this contribution on the total power consumption compared to MRAM. In 2012 and 2017, the largest component to the write power consumption, is switching of the cell. However, as this component is greatly reduced when scaling down the cell, we have estimated for 2017 that charging of the bit lines will be of the same order of magnitude.

We also see in figure 5.6 that the charging of the bit lines consumes considerably more power than the charging of the word lines. This is because of what is mentioned earlier, that the power to charge the bit lines is scaled with the factor N_{ws}, which in this case is assumed to be 16. For the STT-MRAM, the voltage applied to the word line is assumed to be smaller than the voltage applied to the bit line.

5.6.6 PCRAM

For PCRAM, we see in figure 5.6 that the dominating factor to the write power, both in 2012 and 2017, is the switching of the cell. What is special about the PCRAM, is that this contribution become a considerably more dominating factor in 2017 compared to 2012. The reason for this is the multi-level cell which is assumed by the ITRS to be introduced in 2017 [14]. The introduction of multi-level cells has several implications:

- Shorter interconnects are needed, since a smaller array can hold the same amount of data. This reduces the contributions to the total write power from the interconnects.
- Reduces the amount of access transistors connected to one word line, which reduces the contributions of the access transistors to the total power consumed when writing.
- A write and verify algorithm is needed to write the cell. This algorithm increases the amount of energy delivered to the phase change cell.
- During the write and verify algorithm, the pulses start out relatively low, and then gradually increases. However, the same pulse amplitude is applied to the word line during each step of the write and verify algorithm. Therefore, the power consumed to charge the word line is increased compared to the power consumed to charge the bit line.

5.7 Read Power Consumption

The estimated read power for the different memory technologies are shown in figure 5.7. This is the power consumed according to equation (3.15), which represents the sum of the power required to:

- Charge the capacitance of the word lines.
- Charge the capacitance of the bit lines.

- Read the cells.
- Switch the access capacitors.

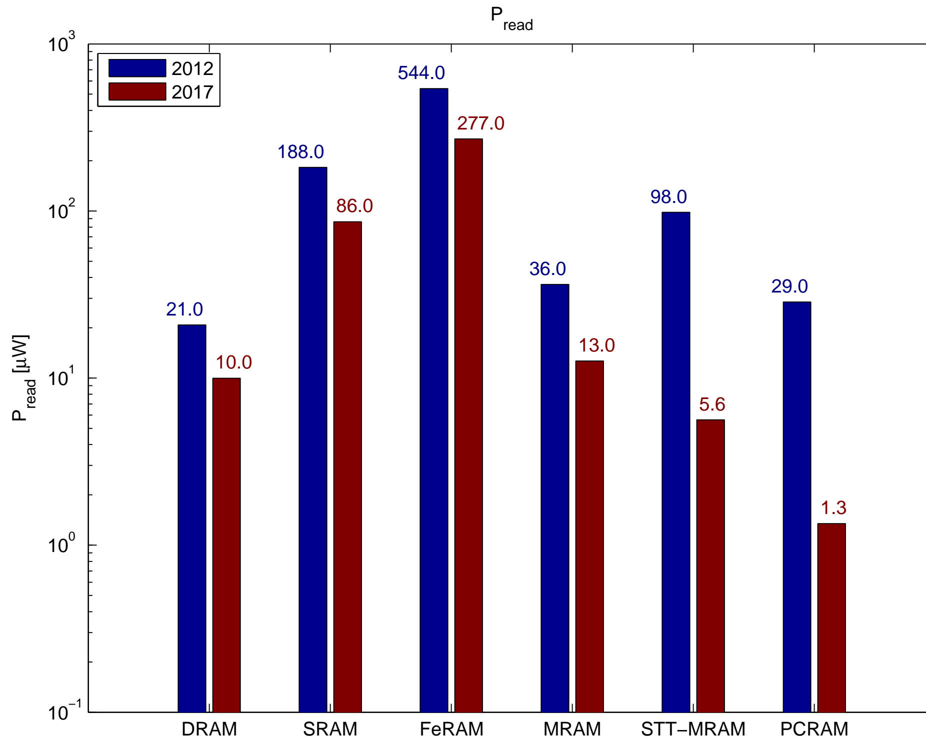


Figure 5.7: The read power of the different memory technologies, shown on a logarithmic scale. The numbers above each bar represent the height of the corresponding bar.

When comparing the read power, shown in figure 5.7, to the write power, shown in figure 5.5, we see that the DRAM, SRAM and FeRAM requires approximately the same amount of power for reading and writing. However, the resistive memories, MRAM, STT-MRAM and PCRAM, have considerably reduced read power compared to write power. This causes the resistive memories to be more energy efficient when reading than the SRAM and FeRAM, while they are at the same order of magnitude as the DRAM.

Similar to the write power, we have broken the read power down into its respective terms, this is shown in figure 5.7. Here the read power breakdown for 2012 is shown in figure 5.8a while the breakdown for 2017 is shown in figure 5.8b.

As we see in figure 5.7, the read power for the resistive memories are reduced because a considerable lower energy is required to read the cell compared to switching

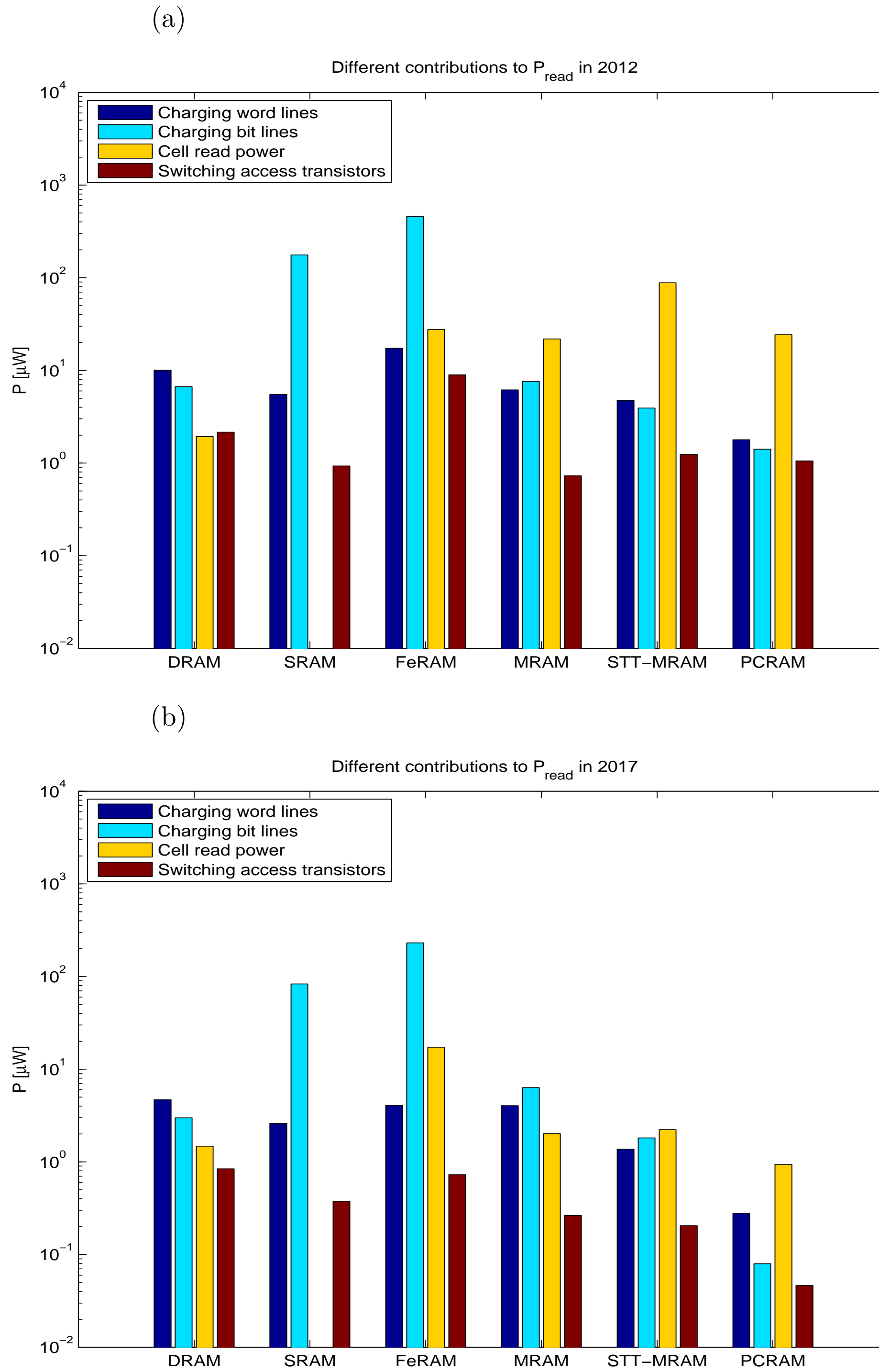


Figure 5.8: Breakdown of read power consumption, shown on a logarithmic scale. (a) Shows the results for 2012. (b) Shows the results for 2017.

it. This also allows for reduced voltages applied to the bit lines, which also reduces the power consumption. The STT-MRAM have the highest cell read power, this is because it has the overall lowest sum of resistance values: $R_{\text{cell}} + R_{BL} + R_{FET}$, where R_{cell} is the average resistance in one cell when reading.

For the capacitive memories, DRAM and FeRAM, a read operation is very similar to a write operation when it comes to power. Because they have a destructive read-out and thus needs to be written back after each read operation. The SRAM requires no energy to read the cell, as no switching occurs there. However, as we see in figure 5.8 a similar voltage needs to be applied to the bit line when reading as when writing. This effect keeps the consumed read power for the SRAM high.

We also see that the read power of the PCRAM is reduced with multi-level cells. The power dissipated in the cell stay almost the same with MLC, even with a sequential read algorithm. This is because we read multiple bits at the same time. However, the power consumed in the interconnects and the access transistors are reduced, this is because shorter interconnects and fewer access transistors are needed.

5.7.1 Relative Signal Strength

The theoretical model does not take into account the energy cost of converting the signal developed on the bitline during reading to digital information. As discussed, this depends on the sense amplifier design, and the sense amplifier designs also vary for the different realization of the same memory technology. However, as a more complex sense amplifier is needed if the signal to be read is small [13], we have estimated the relative signal strength with equations (3.49) and (3.50). The results are showed in figure 5.9, where the signal strength is plotted on a linear scale in blue for 2012 and red for 2017.

As we see in figure 5.9, the maximum relative signal is observed for PCRAM in 2012 and is 0.98, while the lowest is for PCRAM in 2017, when it is 0.15. The large signal strength observed for single-level cells in 2012 is one of the reasons why multi-level cells are a viable option. However, as we see, multi-level cells have a large impact on the relative signal strength, and are expected to require more complex sense amplifiers.

Looking at the other memory technologies, we see that SRAM has the second highest signal strength. One of the reasons for this is that the signal strength for SRAMs depends very much on the design, and the signal strength we have estimated is the upper limit before a read failure occurs [25].

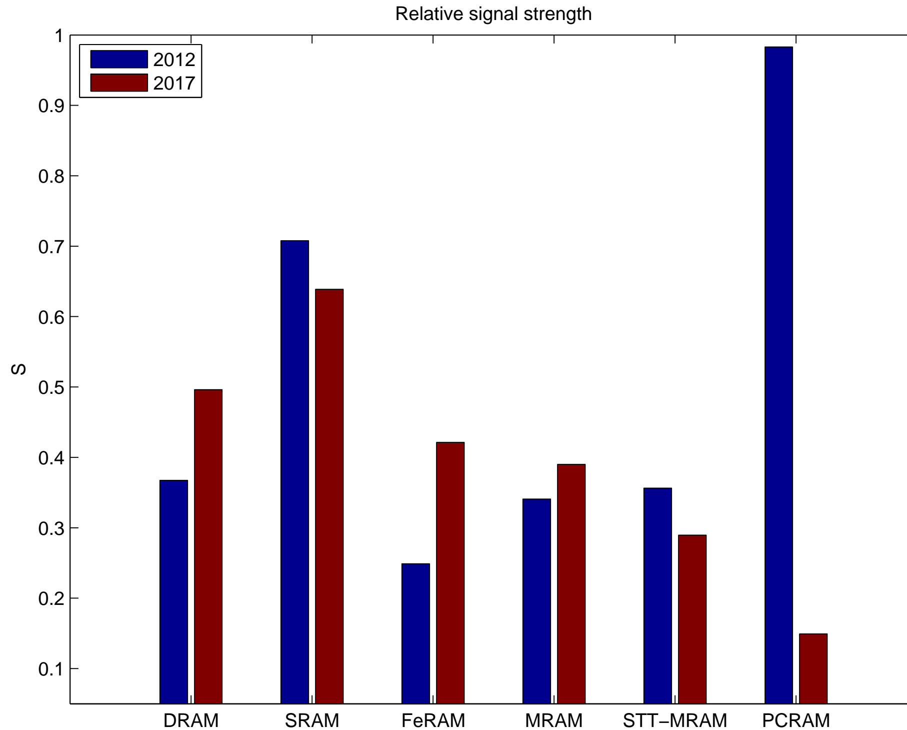


Figure 5.9: The relative signal strength, shown on a linear scale, for the different memories in the model.

The other memories have a signal strength in the range from 0.25 to 0.5, with FeRAM in 2012 as the lowest in this model when excluding the multi-level cell PCRAM. It is important to note that these results are just estimates of the relative signal strength and that differences may occur with different realizations of each memory technology.

5.8 Retention Power Consumption

The DRAM and SRAM requires a constant power supply when inactive, this power consumption is estimated with equations (3.21) and (3.25) respectively. The non-volatile memories can be switched off when inactive and does not require any power to retain data. However, the non-volatile memories can require to be supplied some energy when they are being powered up, as discussed in section 4.3.3. However, as to the author's best knowledge, the power up time is a function of the peripheral circuitry, and not the memory technology. Instant power on, or wake up time in

the order of $\sim$ nanoseconds have been reported for both embedded FeRAM and STT-MRAM [30, 130, 131]. Regardless, the zero passive power can be interpreted as the limit where the passive time is much longer than the wake up time. In this limit, the power consumed during the wake up can be neglected.

The DRAM refresh energy, calculated with equation (3.21), is 70 nW in 2012 and 35 nW in 2017. We have assumed a refresh algorithm where entire rows are refreshed at the same time [15]. This saves much energy, because it is not needed to charge the capacitance of the word line for all the cells.

There exist multiple refresh algorithms that checks if a row recently has been read or written. If that is the case, it does not refresh that row [83]. This reduces the energy needed for refresh when the memory is active over longer periods. However, it requires a larger overhead in terms of control logic etc. Our model does not take any refresh algorithms into account, and will therefore overestimate the refresh power needed if the memory recently has been active.

The SRAM static power consumption calculated with (3.25) is 250 nW in 2012 and 120 nW in 2017. This is a factor of ~ 3.5 larger than the refresh power estimated for DRAM. The assumption we made that have the largest impact on the SRAM leakage current is that low standby power (LSTP) transistors are used. As we see in table 2.4 on page 29, assuming another MOSFET design path would exponentially increase the leakage currents. If low operating power (LOP) transistors were assumed, this would increase the leakage currents by two orders of magnitude [14], while if transistors for high performance were assumed, it would increase the leakage currents by four orders of magnitude [14]. It is important to note that when assuming low standby power transistors we limit the maximum operating frequency of the transistors by a factor of four, as also seen in table 2.4 [14].

The SRAM leakage current is also proportional to the gate width. If the gate width can be reduced to from $3L_g$ to $2L_g$ this will reduce the static power consumption with 33 %.

The SRAM retention power is assumed to scale linearly with the number of cells, as this is directly proportional with the number of transistors. For the DRAM, the refresh power does also scale with the memory size, but instead of with the number of cells it scales almost linearly with the number of memory blocks. This can be understood by looking at equation (3.21), where we see that the DRAM refresh power depends on the number of bit lines and word lines of one block as well as the capacitances of the bit lines and word lines. These parameters are constant if the memory is expanded in blocks of 32 kB (there will be an increase in peripheral circuitry). However, if the block size is increased the DRAM refresh power is

increased almost quadratically, because of the increase in the number of cells and increase in the parasitic capacitances, as can be seen by equation (3.21).

5.9 Power vs Duty Cycle

As seen earlier, the conventional DRAM and SRAM has generally a lower write and read power than the emerging memory technologies (except for the read power of the PCRAM). However, DRAM and SRAM are volatile and must be constantly supplied power in order to retain data. The emerging memory technologies are non-volatile, meaning that they could be turned off when inactive. To visualize this effect, the power is plotted against the duty cycle of the memory in figure 5.10. The duty cycle is defined as:

$$\text{Duty cycle} = \frac{t_{\text{active}}}{T} \tag{5.1}$$

where t_{active} is the time the memory is active and $T = t_{\text{active}} + t_{\text{passive}}$ is the period, this is the same definition as in equation (4.1).

The power vs duty cycle is plotted in figure 5.10a for 2012 and in figure 5.10b for 2017. 40 % writing and 60 % reading are assumed in the active period. This is assumed because the amount of writing is generally lower than the reading for a microcontroller, because there is no point in writing something you are not going to read later [2]. We see that the DRAM is more energy efficient than SRAM for all duty cycles, as it has lower active and passive power.

The points of intersection between the DRAM or SRAM line and the line for the different non-volatile technologies marks the duty cycle where the non-volatile memories have an equal power consumption, as DRAM or SRAM respectively. For duty cycles higher than the point of intersection, DRAM or SRAM is the most energy efficient, while at duty cycles below the respective intersection, the respective non-volatile memory technology are the more energy efficient than the DRAM or SRAM.

As seen in figure 5.10a, DRAM is more energy efficient than the non-volatile memories at duty cycles higher than about 1.6×10^{-4}, which is where the DRAM power intersects the line of the FeRAM power. While SRAM is more energy efficient than the non-volatile memories at duty cycles higher than about 8.5×10^{-4}, which is where the SRAM power intersects the line of the FeRAM power. We also see, in figure 5.10a, that the lines of the FeRAM, STT-MRAM and PCRAM are almost on top of each other when 40 % writing and 60 % reading during the active time, is assumed. The MRAM however, has a considerably higher power consumption

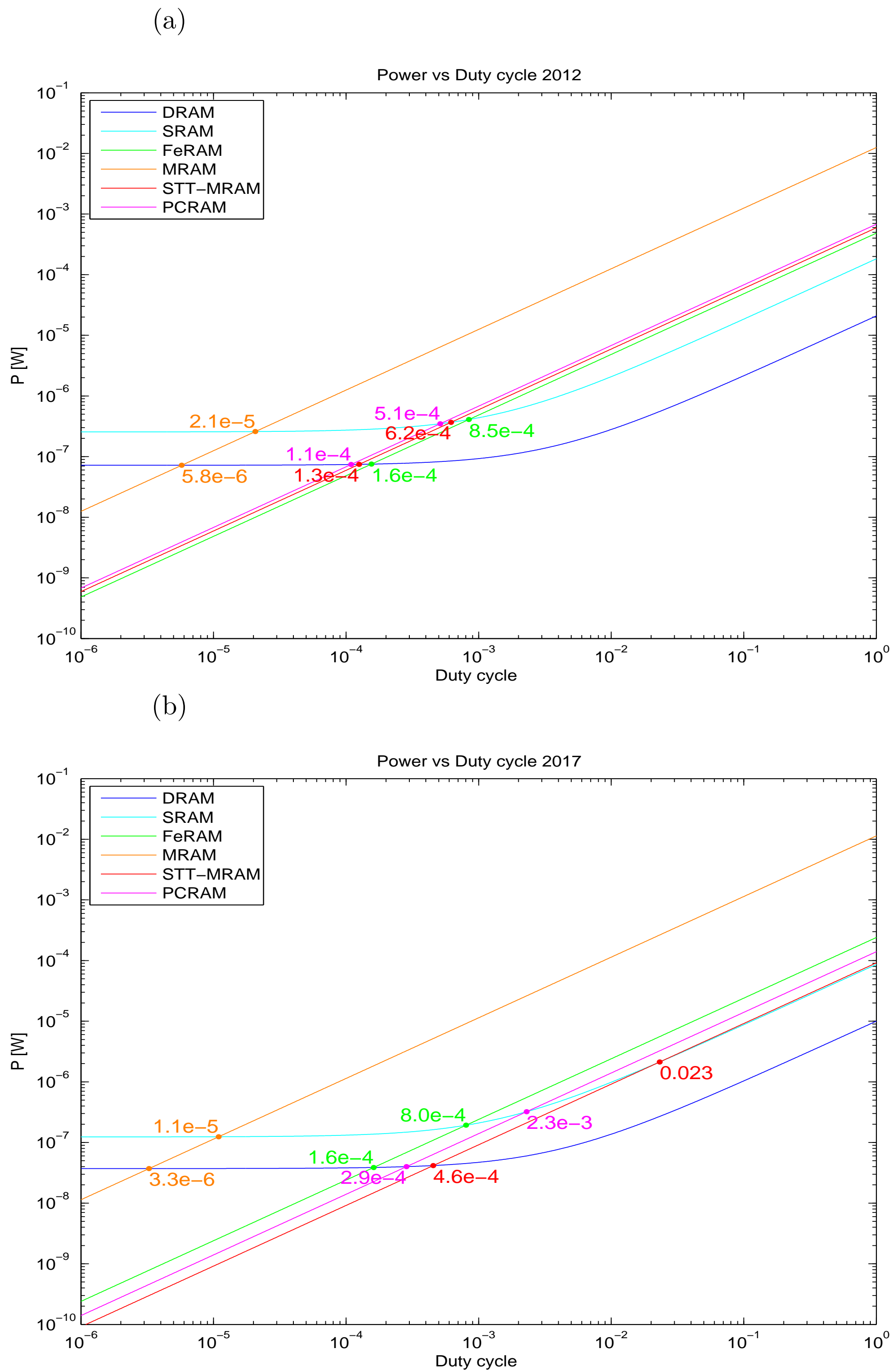


Figure 5.10: Power vs duty cycle. (a) Shows the results for 2012. (b) Shows the results for 2017. The numbers at the points of intersection represents the value of the duty cycle where the curves intersect.

and intersects the DRAM at a duty cycle of 2.1×10^{-5}, and SRAM at a duty cycle of 5.8×10^{-6}.

Figure 5.10b shows the power vs duty cycle for 2017. Comparing with the results from 2012(shown in figure 5.10a), we see that the points of intersection for STT-MRAM and PCRAM have moved to the right. This is because of the large reduction in the estimated STT-MRAM and PCRAM write powers from 2012 to 2017, which are approximately a factor of five. The FeRAM power reduction in power is approximately 2 from 2012 to 2017, this is similar to the reduction in passive power for DRAM and SRAM in the same period. These results explain that the FeRAM intersection points are at approximately the same position in 2012 and 2017. The conventional MRAM shows only a small reduction in the write power when scaled down. Therefore, the points of intersection for the MRAM have moved to the left.

The point of intersection between DRAM and STT-MRAM in 2017 is estimated to be a duty cycle of 4.6×10^{-4}, while the STT-MRAM and SRAM intersects at $2.3 \times 10^{-2} = 0.023$. These duty cycles represents the limits where one of the non-volatile memories have an estimated lower power consumption than the conventional memories in 2017. On the other end of the scale, the intersection point between MRAM and DRAM is at a duty cycle of 3.3×10^{-6}. While the point between MRAM and SRAM is at a duty cycle of 1.1×10^{-5}.

The power consumed by the memory, when powering up, is not taken into account by this model. However, these results are still valid assuming that the passive time is much longer than the power up time.

We have assumed low standby power transistors for the SRAM. If not low standby power transistors are used for SRAM, the exponential increase in leakage current can move the intersection points considerable to the right. This would cause the non-volatile memories to be more energetically favourable than SRAM, for duty cycles in the order of $1 \times 10^{-2} \sim 1 \times 10^{-1}$.

As mentioned, the passive power for the volatile memories depends on the size of the memory. The same is valid for the active power of all the memories, as longer interconnects will result in increased capacitance of the interconnects. However, if the memory is divided into blocks, which makes it possible to segment this capacitance in trade-off for an increased peripheral circuity [13], the active power does not increase as much as the passive power. This means that for larger memories the intersection points would move to the right (i.e. at a higher duty cycle).

The active power is also proportional to the clock frequency and almost proportional to the word size. So, for a higher frequency or word size the intersection

points will be shifted to the left. The amount of reading and writing could also shift the curves, depending on the write and read power shown in figures 5.5 and 5.7 respectively. The other assumptions which are mentioned earlier, such as the sense amplifier, could also impact the points of intersection.

5.10 Iso-Feature Size

When we have collected the parameters for the memory technologies from the ITRS for a given year, the feature size for technology varies considerably. E.g. in 2012 DRAM is assumed to be manufactured with a feature size of $F = 31\,\mathrm{nm}$, while FeRAM is assumed to be manufactured with $F = 180\,\mathrm{nm}$ [14]. To get more insight into how the memory technologies differ we have used the same models as earlier to compare the memories, if they are manufactured with the same feature size. Another reason why iso-feature size is important, is that if a semiconductor company is to select an embedded memory technology, the feature size of the memory will probably be similar to the rest of the chip [2], not decided by of what is possible for standalone memories. To do this we selected the smallest feature size, which is represented in the roadmap, for all the memory technologies discussed in this work. This feature size was found to be 65 nm [14].

A feature size of 65 nm is expected to first be available for MRAM in 2016, and FeRAM in 2021 [14]. For the DRAM, 65 nm was available in 2007 [105], for SRAM, PCRAM and STT-MRAM it was available in the years 2008, 2009 and 2012 respectively [14, 105, 114]. To be consistent, we have assumed that all parameters of the memories are tied to the feature size. This is not necessarily true, as it is possible for example to make multilevel-cells for PCRAM with 65 nm or larger technology [70, 97].

We will in the following only discuss the most relevant results of the model, as all the assumptions of the model already are thoroughly discussed in the results for 2012 and 2017.

5.10.1 Area

The area consumption if the memories were made with 65 nm technology is shown in figure 5.11. As expected, when the memories are manufactured with the same number feature size, the bit densities are closer to each other. As given by equation (3.1), the only parameters that now determines the density is the area factor, X_{AF}.

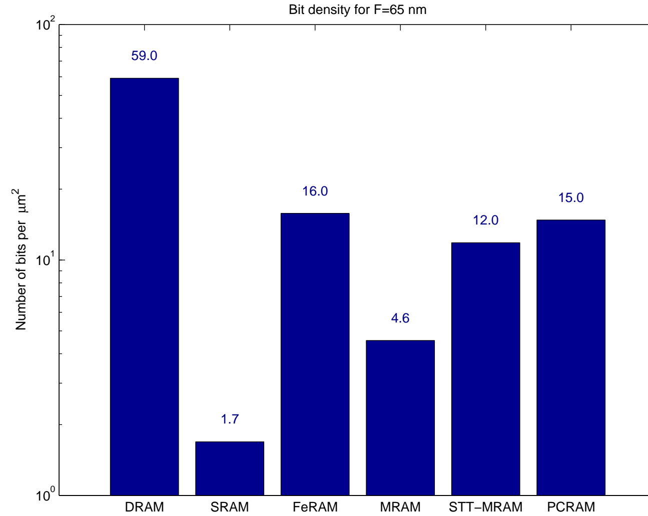


Figure 5.11: The area consumption of the different memory technologies for $F = 65\,\text{nm}$ in terms of bits per μm^2, shown on a logarithmic scale. The numbers above each bar represent the height of the corresponding bar.

As we see in figure 5.11, DRAM is expected to have the lowest area factor $X_{AF} = 6$, for $F = 65\,\text{nm}$. This result in the highest bit density, and the DRAM is followed by FeRAM, PCRAM, STT-MRAM, MRAM respectively. The SRAM have lowest density because of the highest area factor, 140.

5.10.2 Write and Read Power Consumption

In figure 5.12 the write and read powers for $F = 65\,\text{nm}$ are shown, this is the power estimated by equations (3.14) and (3.15) respectively. The write power is shown in blue while the read power is shown in red.

As we see in figure 5.12, the DRAM is estimated to have the lowest write power consumption, while MRAM is expected to have the lowest read power consumption. However, as we see, the MRAM is also expected to have the highest write power, one order of magnitude higher than all of the other memories. The most notable factor that have changed when the memories have the same feature size is that FeRAM have become more energy efficient than the SRAM.

The differences in write and read power can be explained by looking at figure 5.13, here the different terms of equations (3.14) and (3.15) are shown on a logarithmic scale. The write power breakdown is shown in figure 5.13a, while the read power

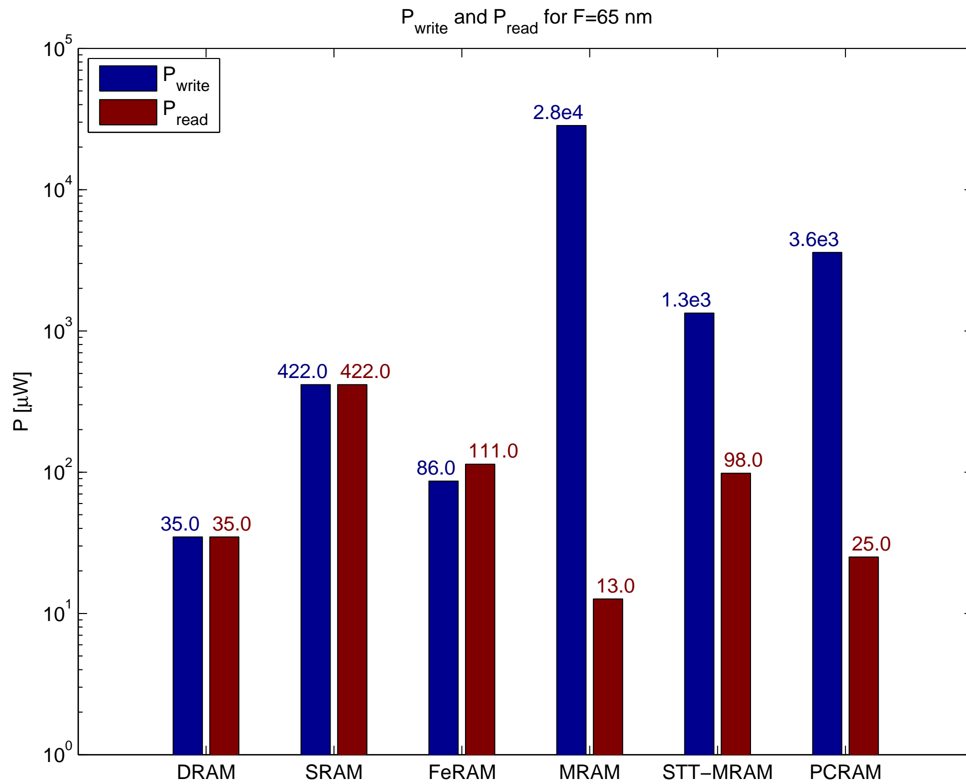


Figure 5.12: The write and read power of the different memory technologies for $F = 65\,\text{nm}$. The numbers above each bar represent the height of the corresponding bar.

breakdown is shown in figure 5.13b.

When we look at the write power breakdown in figure 5.13a, we see that the charging of the interconnects also are almost proportional to the density, leaving SRAM with a relatively high write power consumption. For the resistive memories it is still the switching of the cells which are the dominant factors. For FeRAM, the charging of the interconnects have become more important when scaling down the cell to $F = 65\,\text{nm}$, compared to $F = 180\,\text{nm}$ and $F = 90\,\text{nm}$, which were the feature sizes for FeRAM in 2012 and 2017 respectively.

In figure 5.13a we see that the reasons why the resistive memories MRAM, STT-MRAM and PCRAM have the highest write power, is because they require the highest amount of power to switch the cell. While the SRAM power are higher than the DRAM and FeRAM power, this is because of longer interconnects, which

(a)

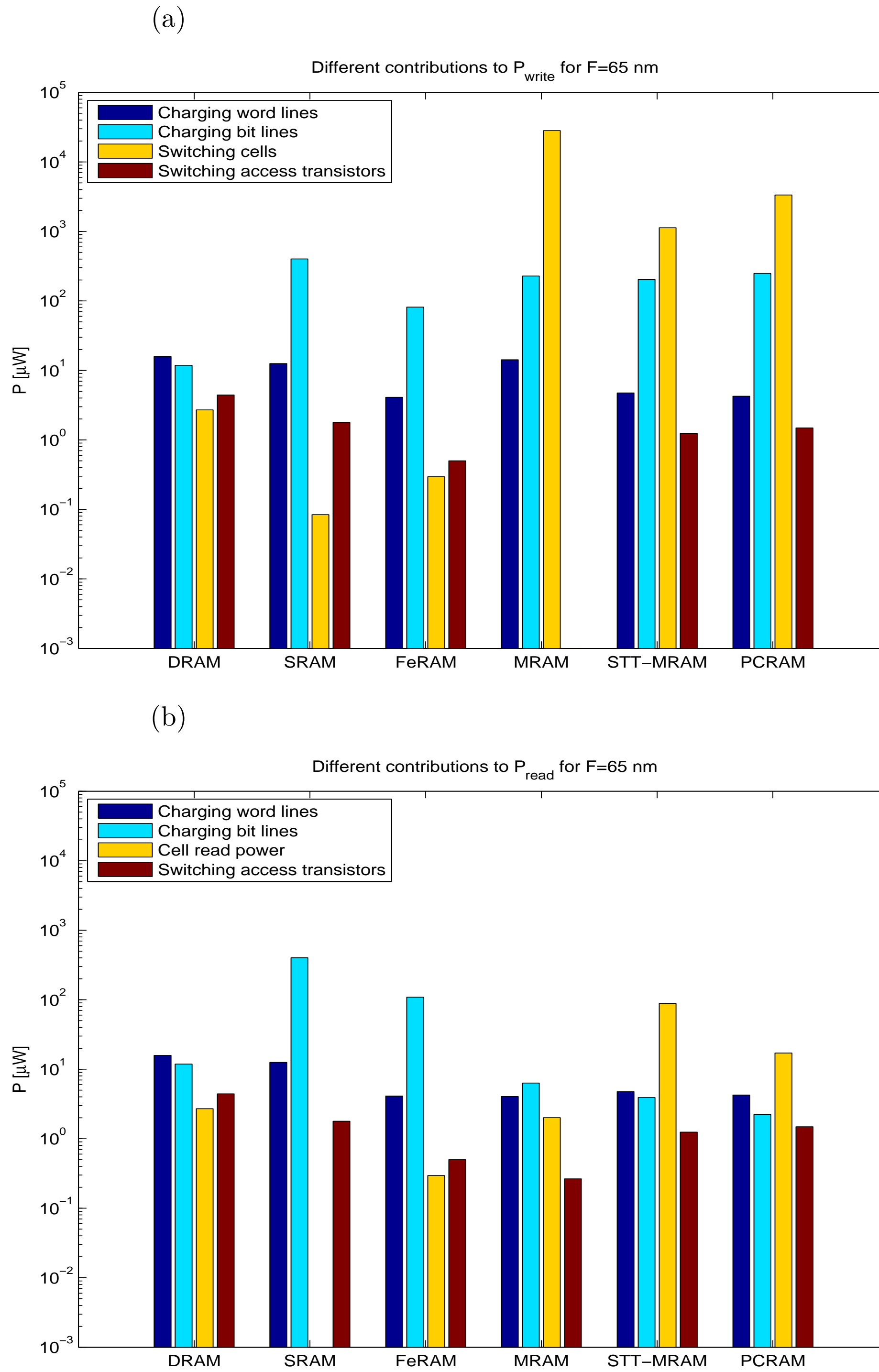


Figure 5.13: Breakdown of write and read power consumption for $F = 65\,\text{nm}$, shown on a logarithmic scale. (a) Write power. (b) Read power.

results in a higher capacitance.

The breakdown of the read power, shown in figure 5.13b, is generally similar to the results for 2012 and 2017. The charge based DRAM, SRAM and FeRAM, spend the majority of the read energy to charge the capacitance of the interconnects. While for the resistive memories there is a considerable amount of energy consumed in the cell, because of resistive effects. In MRAM, the charging of the interconnects are higher than the cell read power, this can be explained by the fact that the conventional MRAM is the least dense of the resistive memories.

5.10.3 Power vs Duty Cycle

We will again illustrate the effect of the non-volatility by plotting the power consumption against the duty cycles of the memories. The duty cycle is defined in equation (5.1). We are still assuming 40 % writing and 60 % reading during the active period. The results are shown in figure 5.14, and again the points of intersection between the lines represents the duty cycle where the non-volatile memories become more energy efficient than the volatile memories.

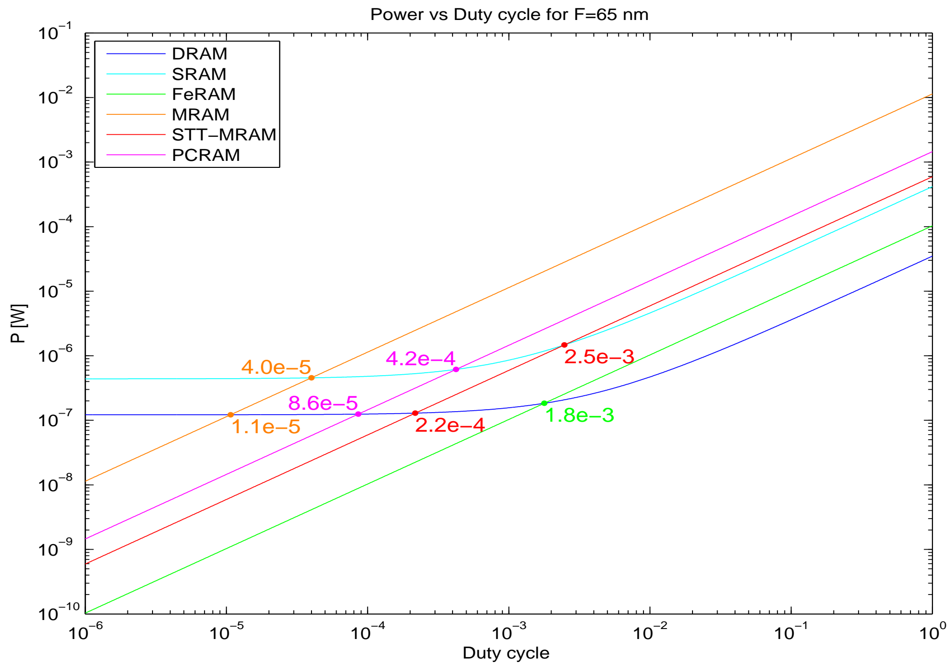


Figure 5.14: Power vs duty cycle for $F = 65\,\text{nm}$. The numbers at the points of intersection represents the value of the duty cycle where the curves intersect.

As we see in figure 5.14, when the feature size is 65 nm, the FeRAM has the lowest power consumption compared to the other non-volatile memory technologies. FeRAM have a lower write and read power consumption than SRAM, meaning that the curves never intersect because FeRAM is more energy efficient for all duty cycles. FeRAM intersects DRAM at a duty cycle of 1.8×10^{-3}, STT-MRAM intersects DRAM at 2.2×10^{-4}, PCRAM intersects DRAM at 8.6×10^{-5}, while MRAM intersects the DRAM line at a duty cycle of 1.1×10^{-5}.

We also see that when the memories are manufactured with similar feature size, the power consumption of conventional MRAM is reduced compared to the other technologies. This results in a shift in the intersecting points between the MRAM and DRAM and the MRAM and SRAM with a factor of about 10, compared to the results for 2012 and 2017.

The shifts for PCRAM and STT-MRAM compared to 2012 and 2017 are not of the same magnitude. The reason for this is that the differences in feature size expected by the ITRS between these two technologies and the volatile technologies are not as large for these technologies compared to MRAM and FeRAM. It is therefore as expected that it would be more beneficial to compare MRAM and FeRAM for the same feature size, instead of for the same year. This is enhanced by the assumption we made that all parameters are tied to the feature size of the memories.

5.11 Important Challenges for the Memory Technologies

This section will briefly mention some of the challenges that the memory technologies have to overcome in order to continue the development towards 2017. We will also mention if there are any other factors that can limit the potential for embedding these technologies on chip.

The main challenges for the development of DRAM are:

- Scaling the equivalent oxide thickness, t_{eq}. An equivalent thickness of 0.3 nm is needed for scaling below the 28 nm node. Implementing a dielectric with $\epsilon_{r,eff} > 50$ is needed [10, 14].
- Reduction of the area factor of the cell down to $X_{AF} = 4$. When scaling down further, the implementation of the $4F^2$ cell becomes very expensive. Vertical channel access transistors (VCAT) are needed, but some challenges still remains [10, 14].
- DRAM is the memory technology with the largest divergence between embedded and standalone versions [33]. This means that the power consump-

tion for the DRAM could diverge significantly from our estimated values if DRAM is embedded on chip [18, 33].

The main challenges for the development of SRAM are:

- Threshold voltage variations are considered the most important challenge [14, 25]. These variations creates a trade-off between yield and area of the memory [25].
- Implementing multi-gate like transistors, like the FinFET, for SRAM. Multi gate transistors reduce threshold voltage variations and leakage currents. However, it is a challenge to implement multi-gate transistors for SRAM applications, because it is no longer possible to finely tune the transistor width to achieve the maximum yield [25, 28].

The main challenges for the development of FeRAM are:

- Because of the destructive read out it is hard to find an electrode material that provides a sufficient amount of remnant polarization, while being stable enough to last for an extended amount of read write/cycles [10, 14]. 1×10^{15} read/write cycles are expected in 2017, while 1×10^{16} is needed to compete with DRAMs in personal computers [13, 14].
- 3D ferroelectric capacitors are needed in order to reduce the area factor X_{AF} to 14. This consists of steep challenges [10, 14].

The main challenge for the conventional MRAM is:

- Because the switching energy does not scale well, the MRAM is not expected to scale beyond the 65 nm node [10, 13, 14].

The main challenges for the development of STT-MRAM are:

- The relative high currents needed to switch the ferromagnetic material can impact the endurance of the magnetic tunnelling junction [10, 14].
- To overcome the problems with thermal noise, perpendicular to the plane magnetization are estimated to be introduced in 2016. However, this requires development of novel materials [10, 14].

The main challenges for the development of PCRAM are:

- Reduction of the reset current is needed in order to allow for a smaller access device [10, 13, 14].
- Reducing the crystallization time to lower write times without impacting other device parameters [64]. The crystallization process is typically in the

order of $\sim 100\,\text{ns}$ [9]. However, crystallization times as low as 1 ns have been shown [64].

- The introduction of four bit per cell in 2017 is considered a large challenge [10, 13, 14]. This will increase the storage density, but will also drastically impact the write and read performance of the device [69, 71].
- The expected endurance is at 1×10^9 write cycles. This is considerably lower than the other memories; however, much effort are put into algorithms that reduces unnecessary write operations [132].
- Increasing the maximum operating temperature above 85 °C [13]. 10 year data retention have been showed for the most common phase change material GST at 85 °C. This temperature is adequate for consumer application, but must be increased for PCRAM to be viable as a memory in e.g. automotive applications. This can be achieved by developing new phase change materials [73, 74].
- The above challenges may reduce the viability of PCRAM as a main memory. However, it is also suggested as a strong competitor to flash memories [11,66].

The main aspect we can take from these points is that all the challenges are concerned about allowing each technology to be scaled further down into the nanometre era. This allows for a lower area, which again is directly proportional to the price per bit [13]. Lower area consumption is also essential to be able to support the memory hungry applications of the future, where up to 90 % of the chip area is expected to be different memory blocks [7, 8].

6 Results and Discussion – Case Studies

In this section we will present and discuss the results from the case studies of the selected memories, listed in table 6.1. As discussed in section 4, we have developed an experimental set-up for the measurements of the memories. However, due to technical difficulties and that a master's degree is only 20 weeks we did not have time to do the actual measurements. We will instead present the estimated power consumption of the memories based on the data given in the data sheets, and then compare these results to the theoretical estimates.

Label	Part number	Memory capacity	Maximum Frequency	Manufacturer	Data sheet reference
FeRAM A	MB85R-256F	256 KBit	6.7 MHz	Fujitsu	[116]
FeRAM B	FM22LD-16	4 Mbit	10 MHz	Ramitron	[117]
MRAM A	MR256A0-8B	256 Kbit	29 MHz	Everspin	[118]
MRAM B	MR2A16A	4 Mbit	29 MHz	Everspin	[119]
PCRAM	NP8P128A-13TSM60E	128 Mbit	8.7 MHz	Micron	[120]
SRAM A	IS61LV256-16AL	4 Mbit	100 MHz	ISSI	[121]
SRAM B	CY62146-EV30	4 Mbit	22.2 MHz	Cypress	[122]

Table 6.1: Selected memory chips and main attributes. The label in the leftmost column represents the labels which are used throughout this work. This is a reprint of table 4.1.

6.1 Active Power Consumption of the Selected Memory Chips

The read and write power consumption of the selected memory chips are shown in figure 6.1. This is the power consumption according to equations (4.7) and (4.8), which again are scaled versions of equations (4.5) and (4.6) respectively. This scaling is done because the active power consumption is approximately proportional

to the operating frequency and the number of I/O ports. Therefore, this acts as a normalizing which allows us to compare the different memories.

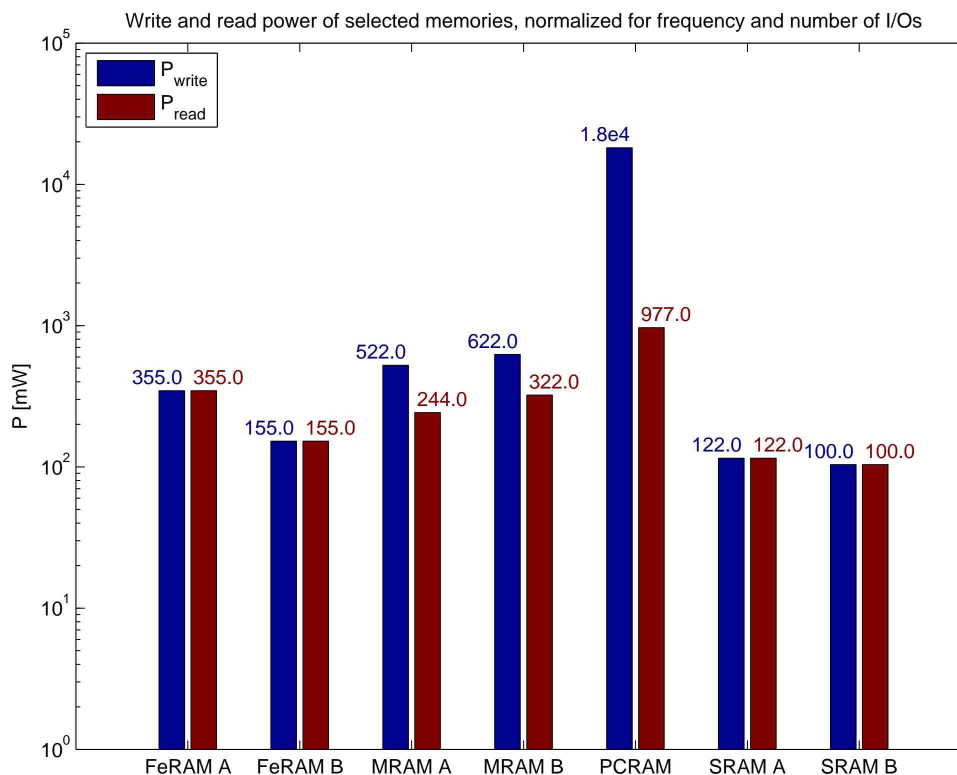


Figure 6.1: The estimated write and read power for the selected memory chips. The power consumption are normalized for frequencies and number of I/O ports. The number above each bar represents the height of the corresponding bar.

In figure 6.1, we see that the two SRAM chips have the lowest write power consumption, followed by the two FeRAMs, then the two MRAMs and finally the PCRAM. We also see that the write powers of all the chips are of the same order of magnitude except for the PCRAM write power which is one order of magnitude higher than the SRAMs. If we exclude the PCRAM, which has a different interface and a considerably larger memory capacity, we see that this is the order as estimated for 2012 and 2017 as shown in figure 5.5.

When looking at the normalized read power, in figure 6.1, we see the same trends and the order of which chips that have the lowest power consumption are approximately the same. However, the MRAM chips have a lower read power consumption than FeRAM A.

We see that while the charge based memories consume similar power for writing

and reading there is a difference for the resistive memories. This is a similar result to those we saw in section 5. We also see that if we exclude the PCRAM, the relative differences between MRAM and the other memories are considerably lower than in section 5. This can be explained by the fact that the theoretical estimates are only done for one memory block, and does not take into account the peripheral circuitry that are needed for a functional memory, like decoders, drivers, sense amplifiers, control logic and connections to the pins. This could in turn wash out the relative differences between the memory technologies. The relative difference between the SRAM and FeRAM is well within same order of magnitude in figure 5.5 and 6.1. We also see that there are some variations from two chips of the same type, as we see when we look at the FeRAM chips.

The absolute values of the power consumption for SRAM and FeRAM are about three orders of magnitude larger than the estimated values, while the values for MRAM are about one order of magnitude larger than the estimated values. This can also be explained by the peripheral circuitry, which contributes with a considerable amount of power that is added to the active power consumed by the memory array. However, it is believed that these contributions are reduced if the memories are embedded on chip, as there is no need for the RAM to spend power on communicating with an external device [8]. Some points which can also contribute with some deviations, is the fact that the theoretical estimates only are done for one memory block of 32 kB (while some of the memory chips have capacities of 4 Mbit = 512 kB). Finally, the values we used to estimate the power for the memory chips are the absolute maximum values given by the data sheets. The reason why the difference between the estimated power consumption and the power consumption in the data sheets for MRAM is lower than the other technologies, can be explained by the fact that MRAM have a considerable higher power consumption in the first place. Thus the large power consumed by the peripheral circuitry will not have the same impact.

Ideally, the memories selected should all have identical characteristics, memory capacity, maximum frequency, number of I/O ports and the same interface. This would make it possible to compare the data sheet values without having to normalize the data, as done with equations (4.5), (4.6), (3.14) and (3.15). However, this was not possible for reasons explained in section 4.1.2. The PCRAM selected have the largest divergence from the estimates, this is because it is manufactured to compete with flash memories, not with SRAM or DRAM [120]. Therefore, the PCRAM has a buffered write interface similar to flash memories; this interface makes it hard to compare the write powers of the PCRAM to the other memories. As a best effort, the typical buffer write time where taken as the cycle time and the full buffer size as the number of I/Os. This gives a picture of the power con-

sumption of this PCRAM chip with that interface, but it is hard to separate what part of the power consumption that can be attributed to the interface, and what part can be attributed to the phase change memory technology.

Generally, we expect the power consumption to increase with memory capacity, as longer interconnects and more peripheral circuitry are needed. This is the case for the MRAM chips, where $P_{\text{MRAM A}} < P_{\text{MRAM B}}$ and the storage density of MRAM A is less than MRAM B. However, this is opposite for the FeRAMs, where $P_{\text{FeRAM A}} > P_{\text{FeRAM B}}$, but the FeRAM B have a higher storage capacity than FeRAM A. This shows that there are more aspects to the memory chips than covered by our simplistic model.

It is also important that the theoretical model is based on the projections of the ITRS, and that any memory device manufacturer is free to deviate from these projections as they see fit. On the other hand, the memories selected for these case studies were selected from the demands discussed in section 4.1.2. For this reason, we can not draw the conclusion that a selected memory, from a given technology, will represent their memory technology as whole. E.g. we cannot say that all MRAMs behave like the two we have selected.

6.2 Passive Power Consumption of the Selected Memory Chips

In this section we will present and discuss the passive power consumption of the selected memories, estimated from equations (4.9) and (4.10).

6.2.1 Volatile Memories

For the SRAMs, we find that the passive power is 54 mW and 25 µW for SRAM A and SRAM B respectively, this is a difference of over 3 orders of magnitude. This can still be explained by the fact that SRAM A are manufactured with high performance (HP) transistors, while SRAM B are manufactured with low standby power transistors (LSTP). As we see in table 2.4, differences up to 4 orders of magnitude are expected between these two transistors design paths.

In table 6.1, we see that the maximum operating frequency of SRAM A are about four times higher than SRAM B, this is consistent with the relative difference in maximum operating frequency between HP and LSTP transistors [14]. If we use the developed SRAM model for SRAM A and B, assuming SRAM A are made with LSTP transistors, and SRAM B are made with HP transistors and that the

memory size is 4 Mbit. We get $P_{\text{static,SRAM A}} \approx 40\,\text{mW}$ and $P_{\text{static,SRAM B}} \approx 4\,\mu\text{W}$, this is within the same order of magnitude for both SRAM chips.

We now see that the estimates for the passive power consumption are considerably better compared to the active power consumption. This fits well with the explanation that it is the peripheral circuitry that substantially contributes to the active power consumption.

6.2.2 Non-Volatile Memories

In the theoretical estimates it was assumed that there was no wake up time of the non-volatile memories, or equivalent that all passive times where much longer than the wake up time. This is not always the case, and we will here assess the consequences of a non-zero wake up time. Although, almost instant power up has been achieved for embedded memories [30, 130, 131].

Because the selected non-volatile memories have a non-zero wake up time, there exists an critical time τ, so that for passive periods longer than τ it is energetically optimal to power down the memories. While for passive time shorter than τ, power can be saved by keeping the memory in standby mode. With the assumptions done in section 4.3.3 this time τ can be expressed as:

$$\tau = \frac{I_{DD,max}}{I_{\text{standby},max}} \cdot t_{\text{power up}} \tag{6.1}$$

Using the data in table 4.8, we can calculate the critical time τ for the selected memories. The results are listed in table 6.2.

	τ [ms]
FeRAM A	10
FeRAM B	27
MRAM A	22
MRAM B	26
PCRAM	26

Table 6.2: Critical passive time for power down. For passive periods longer than τ it is optimal to power down the non-volatile memory.

The selected memories have large differences in standby current and power up time. However, the memories with the longest power up time have the highest standby currents. This results in a critical passive in the order of tens of milliseconds for all the selected memories.

As the power consumed by the memories, when they are powered up, not are a defined parameter in the data sheets, the maximum current where chosen when calculating power up power consumption in equation 4.10. This results can therefore be regarded as an upper limit for the critical passive time.

From this we can conclude: In an energy optimized system, the passive power of the memory chips, given by the standby power, can be calculated with equation (4.9) if the passive period is shorter than τ. If the passive period is longer than τ, the passive power of the non-volatile memories are given by the power used when powering up, and can be calculated with equation (4.10).

6.3 Power vs Duty Cycle

In sections 5.9 and 5.10.3, the theoretical estimated power was plotted against the duty cycles of the memories. We will now do the same, but for the normalized power of the selected memory chips. Here, we have also assumed 40 % writing and 60 % reading during the active period.

An important difference now, when including the wake up time, is that the length of either the active or the passive period becomes a parameter when plotting the power vs duty cycle. We have selected to use the active time as a parameter in the following.

6.3.1 Consequences of a Finite Wake up Time

In figures 6.2a, 6.2b, 6.3a and 6.3b, the normalized power vs duty cycle of the memories are plotted for active times of 100 ms, 1 ms, 10 µs and 100 ns respectively. To explain the quantitatively effects that occurs when including the wake up time of the memories, we will start by discussing figure 6.2a and figure 6.3b, which are the plots active time of 100 ms and 100 ns respectively. This is done as it is easier to understand the effects of the wake up time in the most extreme cases, and these plots represent the limiting situations.

When the active time is 100 ms, as shown in figure 6.2a, the situation looks like what is described in sections 5.9 and 5.10.3. This is because, when the passive time is lower than the critical passive time τ, the duty is so high that the total power consumption is dominated by the active power. Therefore, the small amount of standby power consumed in the passive period has little impact on the total power, and can be neglected. When the duty cycle is reduced, the power consumed during the active period is much more than what is consumed while the memories

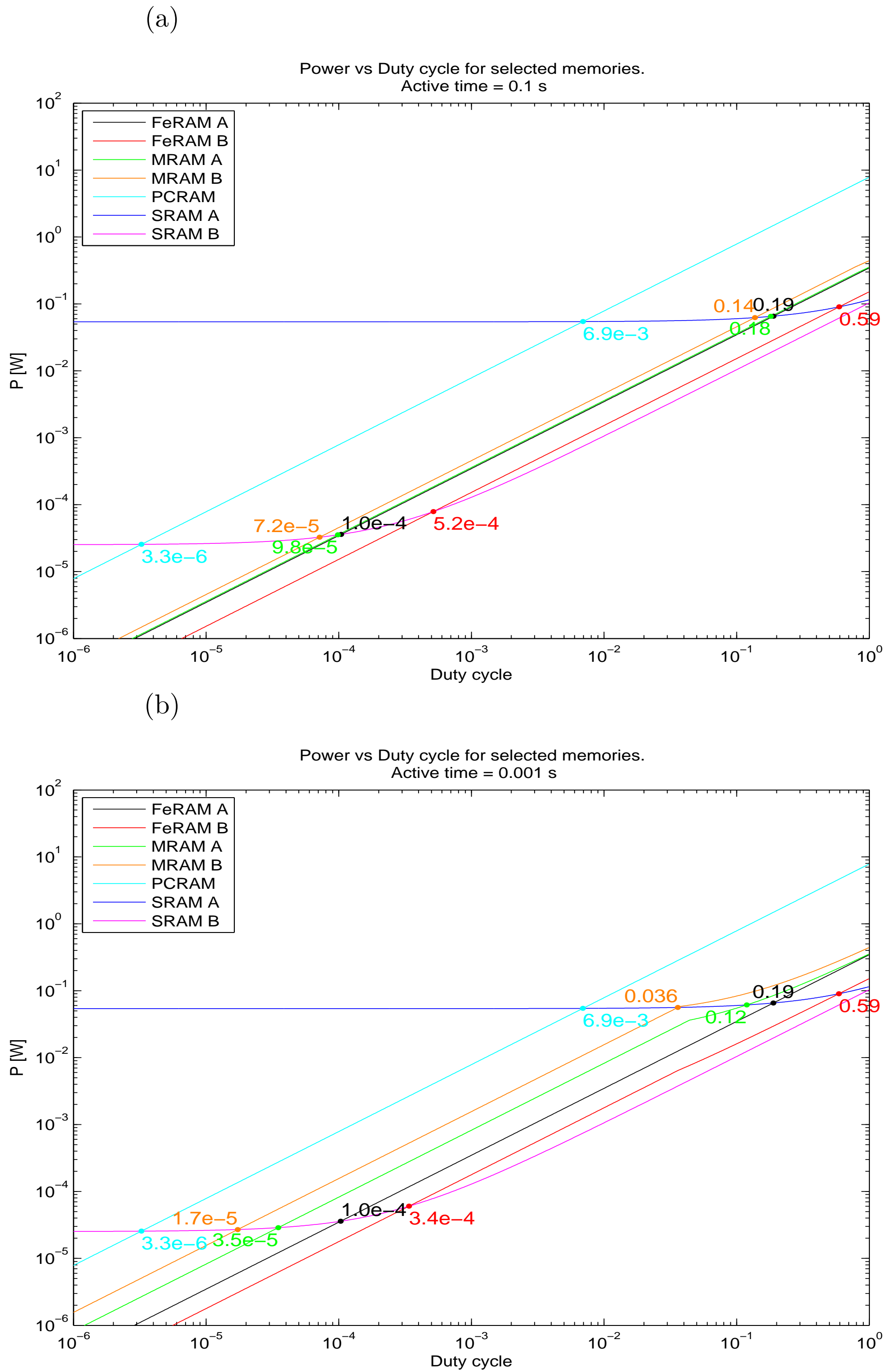


Figure 6.2: Normalized power vs duty cycle. (a) For an active time of 100 ms. (b) For an active time of 1 ms. The numbers at the points of intersection of the non-volatile memories and the SRAMs represents the value of the duty cycle where the curves intersect.

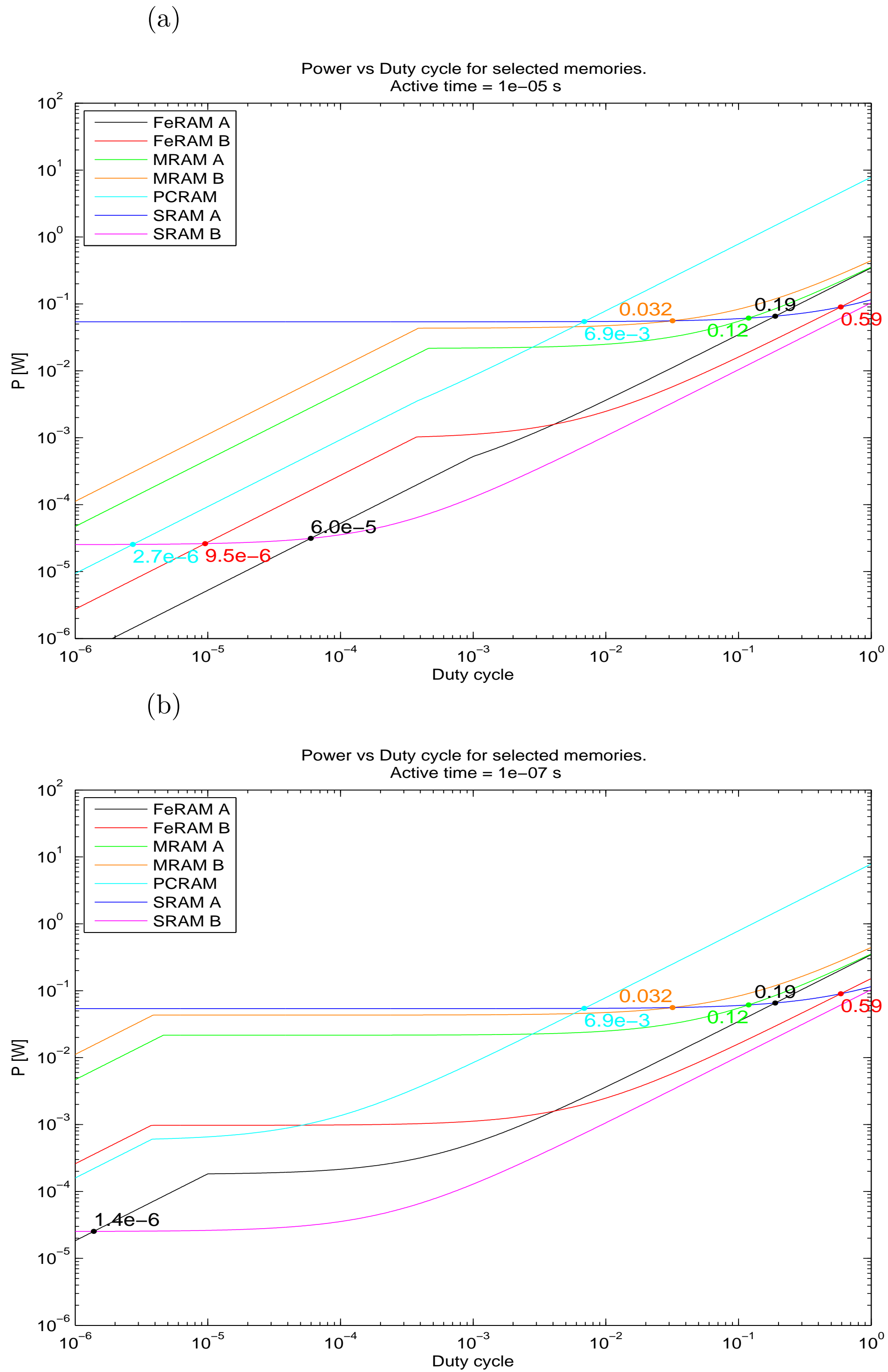


Figure 6.3: Normalized power vs duty cycle. (a) For an active time of 10 µs. (b) For an active time of 100 ns. The numbers at the points of intersection of the non-volatile memories and the SRAMs represents the value of the duty cycle where the curves intersect.

are powered up, this is because of the long active period. Therefore, also the power up contribution can be neglected.

When the active time is 100 ns, as shown in figure 6.3b, all the memories behave like volatile memories for high duty cycles. Volatile memories approach a constant power consumption when the duty cycle is reduced. They continue to behave as volatile memories until they reach the critical duty cycle, which corresponds to the critical passive time τ. In figure 6.3b, this critical duty cycle is in the order of $1 \times 10^{-6} \sim 1 \times 10^{-5}$. After this critical duty cycle is reached, the power is reduced with decreasing duty cycle again, as it is beneficial to power down the memory. The power consumption of the intermediate state, where the power consumption is approximately constant, is proportional to the leakage current of the respective memory.

There is one important thing to note in the case when all the non-volatile memories show a phase where the power consumption is clearly dominated by the standby current, as is the case in figure 6.3b. We see that it is no longer the active power of the non-volatile memories which decides which one is the most energy efficient at low duty cycles; it is now the standby power which is important. This causes the PCRAM and the FeRAM A to be the most energy efficient at low duty cycles, when the active time is in the order of 100 ns.

The situation in figures 6.2b and 6.3a represents intermediate situations compared to 6.2a and 6.3b. In these situations, the active period is short enough for some of the memories to show the tendencies were the wake up power is significant, and some of the memories do not show these tendencies.

When the active time is 1 ms, as shown in figure 6.2b, only the two MRAM chips shows a significant change in power vs duty cycle characteristics compared to figure 6.2a. This is because the MRAM chips have a relatively low ratio of active power consumption to standby power consumption.

On the other hand, when the active time is 10 µs, as shown in figure 6.3a, only the PCRAM and the FeRAM A chips does not show a significant change in power vs duty cycle characteristics compared to figure 6.2a. This is because the PCRAM and the FeRAM A have a relatively high ratio of active power consumption to standby power consumption.

6.3.2 Points of Intersection with the SRAM Curves

We will now discuss the points in figures 6.2 and 6.3, where the curves of different non-volatile memories intersect with the curves of the SRAM chips. These points

of intersection represents the point where the respective non-volatile memory chip is more energy efficient than the respective SRAM chip.

For the SRAM A, we see in figures 6.2 and 6.3 that the curve is almost straight with only a slight increase in power consumption for high duty cycles. There are two main reasons for this: One is that the memory is probably made with high performance transistors. It therefore has a relatively high maximum frequency, but also a high standby current. The other reason is an artefact of the normalization process; we only normalized the active power consumption to 32 MHz not the passive power consumption. If the SRAM A were operated at its maximum frequency of 100 MHz, there would be larger increase in power consumption at higher duty cycles.

We will now look at the points of intersection between the non-volatile memories and the SRAM A for an active time of 100 ms, as shown in figure 6.2a. The points of intersection are in the order of 1×10^{-1}, if one excludes the FeRAM B and PCRAM chips which intersects at a duty cycle of 0.59 and 6.9×10^{-3} respectively. The non-volatile memories have a lower standby current than the SRAM A. For this reason, the points of intersection stay approximately at the same duty cycle, independent of active time of the memories. Some variations are still observed for the two MRAMs which have the highest standby currents of the non-volatile memories. Here, we see that the intersections are moved to lower duty cycles when it is no longer optimal to turn off the MRAM chips.

For SRAM B, which is assumed to be manufactured with low standby power transistors, we see in figures 6.2a, 6.2b, 6.3a and 6.3b that the length of the active period has a considerable impact on the points of intersection. In figure 6.2a, the active time is 100 ms, we see that the intersections with SRAM B are qualitatively similar as the theoretical estimated results from 2012, shown in figure 5.10a. It is the active power consumption which decides where the non-volatile memories intersect the SRAM B line. We also see that the points of intersection are for duty cycles of $5 \times 10^{-5} \sim 5 \times 10^{-4}$ for the FeRAM and MRAM chips, while the PCRAM intersects at a duty cycle of 3.3×10^{-6}.

The FeRAM B intersection point is the result that fits best with theoretical estimates, it intersects at a duty cycle of 5.4×10^{-4}. FeRAM was estimated to intersect with SRAM at a duty cycle of 8.5×10^{-4} in 2012 as shown in figure 5.10a, this is a deviation of 36 %. The other results varies more, this can be explained similar to the deviations discussed in section 6.1. The theoretical estimates are only done for the memory array, without the peripheral circuitry. The peripheral circuitry adds a constant active power consumption to the power consumed in the memory block. This alters the total active power consumption and washes out the

relative differences between the memories.

When the time of the active period is decreased, we see that points of intersection are moved to lower duty cycles. The consequences are different for SRAM A and SRAM B, as SRAM B has a lower standby current than the non-volatile memories. This again causes the points of intersection between SRAM B and the non-volatile memories to be more affected than the points of intersection for SRAM A. For SRAM B, the points of intersection move considerably to lower duty cycles when the length of the active time is decreased. How much each point moves depends on the ratio of power up power consumption and standby power, as well as the critical passive time of the non-volatile memory in question, as seen in figures 6.2 and 6.3.

7 Conclusions and Further Work

A model to estimate the energy consumption of random access memory technologies has been made for six different memory technologies, which are all expected to be in production by 2017. The different memory technologies are: DRAM, SRAM, FeRAM, MRAM, STT-MRAM and PCRAM. The focus has been on developing a model to determine which of the memory technologies is the most energy efficient.

The density of the memory is of utmost importance when estimating the power consumption, this because one of the main contributors to the power are the capacitance of the interconnects. The density of the memory is also one of the most important factors for a memory technology, as it is directly proportional to the price per bit [13]. It was shown that in 2012, the conventional DRAM is expected to be the densest memory. However, the PCRAM and STT-MRAM are denser than SRAM. The MRAM and FeRAM are expected to have the lowest densities both in 2012 and 2017. In 2017 the PCRAM is expected to become denser than the DRAM because PCRAM is able to store multiple bits per cell.

The write and read power were estimated for the selected memory technologies. The model for the write (read) power took into account the cell write (read) energy per bit as well as the charging of the parasitic capacitances of the interconnects, and switching of the gate capacitance of the access transistors. When evaluating the estimates it was found that DRAM is expected to have a write power about one order of magnitude lower than the other memories. The SRAM had the second lowest write power, but it was estimated to be of the same order of magnitude as FeRAM. The memories that showed the largest improvements from 2012 to 2017 were STT-MRAM and PCRAM. This is because of the large reduction in switching current for these technologies when scaling down the cell.

When evaluating the read power, it was found that the charge based memories DRAM, SRAM and FeRAM had about the same read power as write power. While the resistive memories, MRAM, STT-MRAM and PCRAM, showed a substantial decrease in read power compared to write power. It was shown that this reduction caused PCRAM to have the lowest read power of the evaluated technologies.

The DRAM and SRAM are volatile; this means that they need to spend power to retain data. This is not needed for the non-volatile technologies, which can be powered down when not in use. With a power vs duty cycle plot it has been shown for what duty cycle the non-volatile memories will be more energy efficient than the DRAM and SRAM. In 2012, the FeRAM had the lowest power consumption of the non-volatile memories, when writing 40 % and reading 60 % of the active time.

The FeRAM was estimated to be more energy efficient than the SRAM for duty cycles lower than 8.5×10^{-4}, and more energy efficient than DRAM for duty cycles lower than 1.6×10^{-4}. For 2017, the STT-MRAM was the most energy efficient of the non-volatile memories, showing a lower power consumption than SRAM for duty cycles lower than 2.3×10^{-2}, and lower than DRAM for duty cycles lower than 4.6×10^{-4}.

The developed model was also used to evaluate the energy consumption and density of the different memory technologies if they were manufactured with the same feature size. $F = 65\,\text{nm}$ was chosen, as this represents the lowest feature size that are available in the ITRS roadmap for MRAM and FeRAM [14]. The area analysis showed, with the assumptions made, that DRAM would still be the densest memory technology followed by FeRAM. The least dense technology in this analysis was the SRAM. This analysis also estimated the FeRAM to be the non-volatile memory technology with the lowest power consumption. FeRAM was shown to have a lower power consumption than SRAM for all duty cycles, and a lower power consumption than DRAM for duty cycles below 1.8×10^{-3}.

A set-up for experimental evaluation of the non-volatile memory technologies have been developed. However, as a master thesis at NTNU is only 20 weeks, there was not enough time to do the actual measurements. The memories selected for the test has instead been evaluated on the performance described in their data sheets. This allowed us to quantitatively explore the effects for a real product, like peripheral circuitry and a finite wake up time for the non-volatile memories. It was shown that the active power consumption of the selected memories were substantially larger than what could be explained by the theoretical model. However, the deviations for the passive power consumption were within the same order of magnitude. This can be explained by the effect of the peripheral circuitry needed for a functional memory, which adds as considerable contributor to the active power consumption. This contribution is believed to be reduced if the memories are embedded on chip.

Future work could consist of expanding the model. This could be done by taking other aspects into account, like sense amplifiers and more of the peripheral circuitry. The model could also be expanded to include other memory technologies like Flash, Racetrack memory, Mott memory, Redox memory, Nano-RAM and Millipede memory. As an experimental set-up has been developed, it would be natural to do the actual measurements now that the technical difficulties regarding the voltage regulator has been solved. Only a few selected memories were selected for the prototype testing of the experimental set-up. It would be necessary to expand the selection to cover more memories in order to draw some valid conclusions about the memory technologies.

References

[1] M. Moreau, "Estimating the energy consumption of emerging random access memory technologies." Project Thesis, Department of Electronics and Telecommunications, December 2012.

[2] A. P. Syvertsen, 2012-2013. Private communication.

[3] P. Marwedel, *Embedded System Design : Embedded Systems Foundations of Cyber-Physical Systems*. Dordrecht: Springer Netherlands, 2011.

[4] E. C. Hall, *Journey to the moon : the history of the Apollo guidance computer*. Reston, Va.: American Institute of Aeronautics and Astronautics, 1996.

[5] G. Moore, "Cramming more components onto integrated circuits (Reprinted from Electronics, pg 114-117, April 19, 1965)," *PROCEEDINGS OF THE IEEE*, vol. 86, pp. 82–85, JAN 1998.

[6] D. Patterson, "Microprocessors in 2020," *Scientific American*, vol. 273, no. 3, pp. 48–51, 1995.

[7] S. Hamdioui, "Testing embedded memories: A survey," in *Mathematical and Engineering Methods in Computer Science*, vol. 7721 of *Lecture Notes in Computer Science*, pp. 32–42, Springer Berlin Heidelberg, 2013.

[8] E. Marinissen, B. Prince, D. Keitel-Schulz, and Y. Zorian, "Challenges in embedded memory design and test," in *DESIGN, AUTOMATION AND TEST IN EUROPE CONFERENCE AND EXHIBITION, VOLS 1 AND 2, PROCEEDINGS* (Wehn, N and Benini, L, ed.), Design Automation and Test in Europe Conference and Expo, (10662 LOS VAQUEROS CIRCLE, PO BOX 3014, LOS ALAMITOS, CA 90720-1264 USA), pp. 722–727, EDDA; EDAC; IEEE Comp Soc, TTTC; IEEE Comp Soc, DATC; ECSI; ACM SIGDA; RAS, IEEE COMPUTER SOC, 2005. Design, Automation and Test in Europe Conference and Exhibition (DATE 05), Munich, GERMANY, MAR 07-11, 2005.

[9] "International Technology Roadmap for Semiconductors(ITRS), Tables: Emerging Research Devices(ERD)," 2011. http://www.itrs.net/Links/2011ITRS/2011Tables/ERD_2011Tables.xlsx.

[10] "International Technology Roadmap for Semiconductors(ITRS), Chapter: Process Integration, Devices, and Structures (PIDS)," 2011. http://www.itrs.net/Links/2011ITRS/2011Chapters/2011PIDS.pdf.

[11] G. W. Burr, M. J. Breitwisch, M. Franceschini, D. Garetto, K. Gopalakrishnan, B. Jackson, B. Kurdi, C. Lam, L. A. Lastras, A. Padilla, B. Rajendran, S. Raoux, and R. S. Shenoy, "Phase change memory technology," *JOURNAL OF VACUUM SCIENCE & TECHNOLOGY B*, vol. 28, pp. 223–262, MAR 2010.

[12] http://www.itrs.net/about.html.

[13] R. Waser, "Memory devices and storage systems," in *Nanoelectronics and Information Technology*, ch. 26-29, pp. 603–681, John Wiley & Sons, 2012.

[14] "International Technology Roadmap for Semiconductors(ITRS), Tables: Process Integration, Devices and Structures(PIDS)," 2012. http://www.itrs.net/Links/2012ITRS/2012Tables/PIDS_2012Tables.xlsx.

[15] A. K. Sharma, *Semiconductor memories : technology, testing, and reliability*, ch. 2, pp. 10–71. Piscataway, N.J.: IEEE Press, 1997.

[16] K. C. Chun, W. Zhang, P. Jain, and C. H. Kim, "A 2T1C Embedded DRAM Macro With No Boosted Supplies Featuring a 7T SRAM Based Repair and a Cell Storage Monitor," *IEEE JOURNAL OF SOLID-STATE CIRCUITS*, vol. 47, pp. 2517–2526, OCT 2012.

[17] "International Technology Roadmap for Semiconductors(ITRS), Chapter System Drivers (SYSD)," 2011. http://www.itrs.net/Links/2011ITRS/2011Chapters/2011SysDrivers.pdf.

[18] "International Technology Roadmap for Semiconductors(ITRS), Tables System Drivers (SYSD)," 2011. http://www.itrs.net/Links/2011ITRS/2011Tables/SysDrivers_2011Tables.xlsx.

[19] U. Böttger and S. R. Summerfelt, *Nanoelectronics and Information Technology*, ch. 22, pp. 565–588. John Wiley & Sons, 2005.

[20] K. Lee and S. H. Kang, "Development of Embedded STT-MRAM for Mobile System-on-Chips," *IEEE TRANSACTIONS ON MAGNETICS*, vol. 47, pp. 131–136, JAN 2011. 21st Magnetic Recording Conference (TMRC 2010), Univ California, San Diego, CA, AUG 16-18, 2010.

[21] D. Cai, H. Chen, Q. Wang, Y. Chen, Z. Song, G. Wu, and S. Feng, "An 8-Mb Phase-Change Random Access Memory Chip Based on a Resistor-on-Via-Stacked-Plug Storage Cell," *IEEE ELECTRON DEVICE LETTERS*, vol. 33, pp. 1270–1272, SEP 2012.

[22] J. Knoch, S. Mantl, and S. Feste, *Nanoelectronics and Information Technology*, ch. 14, pp. 341–373. John Wiley & Sons, 2012.

[23] B. G. Streetman and S. Banerjee, *Solid State electronic devices*, ch. 6, pp. 251–334. Prentice Hall series in solid state physical electronics, Prentice Hall New Jersey, 6th ed., 2006.

[24] C. Kittel, *Introduction to solid state physics*, ch. 8, pp. 185–220. John Wiley & Sons, 8th ed., 2005.

[25] C. Shin, *Advanced MOSFET designs and implications for SRAM scaling.* PhD thesis, University of California, 2011.

[26] A. Asenov, "Simulation of statistical variability in nano MOSFETs," in *2007 Symposium on VLSI Technology, Digest of Technical Papers*, (5TH FLOOR KUDAN KITA BLDG 1-12-3 KUDAN-KITA CHIYODA-KU, TOKYO, 102, JAPAN), pp. 86–87, Japan Soc Appl Phys; IEEE Electron Devices Soc; IEEE Solid-State Circuits Soc, JAPAN SOCIETY APPLIED PHYSICS, 2007. Symposium on VLSI Technology 2007, Kyoto, JAPAN, 2007.

[27] B. G. Streetman and S. Banerjee, *Solid State electronic devices*, ch. Appendix III, p. 540. Prentice Hall series in solid state physical electronics, Prentice Hall New Jersey, 6th ed., 2006.

[28] Z. Jaksic and R. Canal, "Comparison of sram cells for 10-nm soi finfets under process and environmental variations," *Electron Devices, IEEE Transactions on*, vol. 60, no. 1, pp. 49–55, 2013.

[29] http://semimd.com/blog/2011/05/04/intel-going-vertical-for-22nm-transistors/.

[30] M. Qazi, M. Clinton, S. Bartling, and A. P. Chandrakasan, "A Low-Voltage 1 Mb FRAM in 0.13 mu m CMOS Featuring Time-to-Digital Sensing for Expanded Operating Margin," *IEEE JOURNAL OF SOLID-STATE CIRCUITS*, vol. 47, pp. 141–150, JAN 2012.

[31] H. Kawaura and T. Baba, "Direct tunneling from source to drain in nanometer-scale silicon transistors," *JAPANESE JOURNAL OF APPLIED PHYSICS PART 1-REGULAR PAPERS SHORT NOTES & REVIEW PAPERS*, vol. 42, pp. 351–357, FEB 2003.

[32] R. Dennard, "Bit line," June 4 1968. US Patent 3,387,286.

[33] K. Itoh, M. Horiguchi, and H. Tanaka, "Ultra-low voltage nano-scale dram cells," in *Ultra-Low Voltage Nano-Scale Memories* (K. Itoh, M. Horiguchi, and H. Tanaka, eds.), Series On Integrated Circuits And Systems, pp. 79–117, Springer US, 2007.

[34] G. Wang, D. Anand, N. Butt, A. Cestero, M. Chudzik, J. Ervin, S. Fang, G. Freeman, H. Ho, B. Khan, B. Kim, W. Kong, R. Krishnan, S. Krishnan, O. Kwon, J. Liu, K. McStay, E. Nelson, K. Nummy, P. Parries, J. Sim, R. Takalkar, A. Tessier, R. M. Todi, R. Malik, S. Stiffler, and S. S. Iyer, "Scaling Deep Trench Based eDRAM on SOI to 32nm and Beyond," in *2009 IEEE INTERNATIONAL ELECTRON DEVICES MEETING*, International Electron Devices Meeting, (345 E 47TH ST, NEW YORK, NY 10017 USA), pp. 236–239, IEEE Elect Devices Soc, IEEE, 2009. IEEE International Electron Devices Meeting (IEDM 2009), Baltimore, MD, DEC 07-09, 2009.

[35] K. C. Chun, P. Jain, T.-H. Kim, and C. H. Kim, "A 667 MHz Logic-Compatible Embedded DRAM Featuring an Asymmetric 2T Gain Cell for High Speed On-Die Caches," *IEEE JOURNAL OF SOLID-STATE CIRCUITS*, vol. 47, pp. 547–559, FEB 2012.

[36] "International Technology Roadmap for Semiconductors(ITRS), Chapter: Emerging Research Devices(ERD)," 2011. http://www.itrs.net/Links/2011ITRS/2011Chapters/2011ERD.pdf.

[37] K.-W. Song, J.-Y. Kim, J.-M. Yoon, S. Kim, H. Kim, H.-W. Chung, H. Kim, K. Kim, H.-W. Park, H. C. Kang, N.-K. Tak, D. Park, W.-S. Kim, Y.-T. Lee, Y. C. Oh, G.-Y. Jin, J. Yoo, D. Park, K. Oh, C. Kim, and Y.-H. Jun, "A 31 ns Random Cycle VCAT-Based 4F(2) DRAM With Manufacturability and Enhanced Cell Efficiency," *IEEE JOURNAL OF SOLID-STATE CIRCUITS*, vol. 45, pp. 880–888, APR 2010. Symposium on VLSI Circuits, Kyoto, JAPAN, JUN 16-18, 2009.

[38] V. V. Zhirnov, R. K. Cavin, III, S. Menzel, E. Linn, S. Schmelzer, D. Braeuhaus, C. Schindler, and R. Waser, "Memory Devices: Energy-Space-Time Tradeoffs," *PROCEEDINGS OF THE IEEE*, vol. 98, pp. 2185–2200, DEC 2010.

[39] K. Ishibashi, *Low Power and Reliable SRAM Memory Cell and Array Design*, vol. 31, ch. 2, pp. 5–10. Springerverlag Berlin Heidelberg, 2011.

[40] J. Singh, S. P. Mohanty, and D. K. Pradhan, *Robust SRAM Designs and Analysis*, ch. 1, pp. 1–29. Springer, 2012.

[41] M. Abu-Rahma and M. Anis, "Variability in nanometer technologies and impact on sram," in *Nanometer Variation-Tolerant SRAM*, pp. 5–47, Springer New York, 2013.

[42] E. SEEVINCK, F. LIST, and J. LOHSTROH, "STATIC-NOISE MARGIN ANALYSIS OF MOS SRAM CELLS," *IEEE JOURNAL OF SOLID-STATE CIRCUITS*, vol. 22, pp. 748–754, OCT 1987.

[43] K. Osada, "Fundamentals of sram memory cell," in *Low Power and Reliable SRAM Memory Cell and Array Design* (K. Ishibashi and K. Osada, eds.), vol. 31 of *Springer Series in Advanced Microelectronics*, pp. 5–10, Springer Berlin Heidelberg, 2011.

[44] H. Yamauchi, "Embedded sram design in nanometer-scale technologies," in *Embedded Memories for Nano-Scale VLSIs* (K. Zhang, ed.), Integrated Circuits and Systems, pp. 39–88, Springer US, 2009.

[45] S. Hoffmann-Eifert, D. Richter, and S. T.-M. Kinstry, *Nanoelectronics and Information Technology*, ch. 1, pp. 31–61. John Wiley & Sons, 2012.

[46] D. Takashima, Y. Nagadomi, and T. Ozaki, "A 100 MHz Ladder FeRAM Design With Capacitance-Coupled-Bitline (CCB) Cell," *IEEE JOURNAL OF SOLID-STATE CIRCUITS*, vol. 46, pp. 681–689, MAR 2011.

[47] A. Bune, V. Fridkin, S. Ducharme, L. Blinov, S. Palto, A. Sorokin, S. Yudin, and A. Zlatkin, "Two-dimensional ferroelectric films," *NATURE*, vol. 391, pp. 874–877, FEB 26 1998.

[48] D. Fong, G. Stephenson, S. Streiffer, J. Eastman, O. Auciello, P. Fuoss, and C. Thompson, "Ferroelectricity in ultrathin perovskite films," *SCIENCE*, vol. 304, pp. 1650–1653, JUN 11 2004.

[49] D. Bürgler and P. Grünberg, *Nanoelectronics and Information Technology*, ch. 4, pp. 112–129. John Wiley & Sons, 2012.

[50] http://www.everspin.com/technology.php?qtype=3.

[51] L. Savtchenko, A. A. Korkin, B. N. Engel, N. D. Rizzo, M. F. Deherrera, and J. A. Janesky, "Method of writing to scalable magnetoresistance random access memory element," Apr. 8 2003. US Patent 6,545,906.

[52] D. Worledge, "Spin flop switching for magnetic random access memory," *APPLIED PHYSICS LETTERS*, vol. 84, pp. 4559–4561, MAY 31 2004.

[53] R. Sousa and I. Prejbeanu, "Non-volatile magnetic random access memories (MRAM)," *COMPTES RENDUS PHYSIQUE*, vol. 6, pp. 1013–1021, NOV 2005.

[54] M. Stiles and A. Zangwill, "Anatomy of spin-transfer torque," *PHYSICAL REVIEW B*, vol. 66, JUL 1 2002.

[55] http://www.everspin.com/technology.php?qtype=4.

[56] Y. Huai, M. Pakala, Z. Diao, and Y. Ding, "Spin transfer switching current reduction in magnetic tunnel junction based dual spin filter structures," *APPLIED PHYSICS LETTERS*, vol. 87, NOV 28 2005.

[57] http://www.everspin.com/technology.php?qtype=5.

[58] J. Kim, T. Kim, W. Hao, H. Rao, K. Lee, X. Zhu, X. Li, W. Hsu, S. Kang, N. Matt, and N. Yu, "A 45nm 1mb embedded stt-mram with design techniques to minimize read-disturbance," in *VLSI Circuits (VLSIC), 2011 Symposium on*, pp. 296–297, 2011.

[59] T. Kawahara, "Scalable spin-transfer torque ram technology for normally-off computing," *Design Test of Computers, IEEE*, vol. 28, no. 1, pp. 52–63, 2011.

[60] I. L. Prejbeanu, S. Bandiera, J. Alvarez-Herault, R. C. Sousa, B. Dieny, and J.-P. Nozieres, "Thermally assisted MRAMs: ultimate scalability and logic functionalities," *JOURNAL OF PHYSICS D-APPLIED PHYSICS*, vol. 46, FEB 20 2013.

[61] W. Kim, J. H. Jeong, Y. Kim, W. C. Lim, J. H. Kim, J. H. Park, H. J. Shin, Y. S. Park, K. S. Kim, S. H. Park, Y. J. Lee, K. W. Kim, H. J. Kwon, H. L. Park, H. S. Ahn, S. C. Oh, J. E. Lee, S. O. Park, S. Choi, H. K. Kang, and C. Chung, "Extended scalability of perpendicular STT-MRAM towards sub-20nm MTJ node," in *2011 IEEE INTERNATIONAL ELECTRON DEVICES MEETING (IEDM)*, (345 E 47TH ST, NEW YORK, NY 10017 USA), IEEE; IEEE Electron Devices Soc, IEEE, 2011. IEEE International Electron Devices Meeting (IEDM), Washington, DC, DEC 05-07, 2011.

[62] M. Gajek, J. J. Nowak, J. Z. Sun, P. L. Trouilloud, E. J. O'Sullivan, D. W. Abraham, M. C. Gaidis, G. Hu, S. Brown, Y. Zhu, R. P. Robertazzi, W. J. Gallagher, and D. C. Worledge, "Spin torque switching of 20 nm magnetic tunnel junctions with perpendicular anisotropy," *APPLIED PHYSICS LETTERS*, vol. 100, MAR 26 2012.

[63] T. Kawahara, K. Ito, R. Takemura, and H. Ohno, "Spin-transfer torque RAM technology: Review and prospect," *MICROELECTRONICS RELIABILITY*, vol. 52, pp. 613–627, APR 2012.

[64] D. Loke, L. Shi, W. Wang, R. Zhao, H. Yang, L.-T. Ng, K.-G. Lim, T.-C. Chong, and Y.-C. Yeo, "Ultrafast switching in nanoscale phase-change ran-

dom access memory with superlattice-like structures," *NANOTECHNOLOGY*, vol. 22, JUN 24 2011.

[65] B. C. Lee, P. Zhou, J. Yang, Y. Zhang, B. Zhao, E. Ipek, O. Mutlu, and D. Burger, "PHASE-CHANGE TECHNOLOGY AND THE FUTURE OF MAIN MEMORY," *IEEE MICRO*, vol. 30, pp. 131–141, JAN-FEB 2010. 36th Annual International Symposium on Computer Architecture, Austin, TX, JUN 20-24, 2009.

[66] 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.

[67] M. Joshi, W. Zhang, and T. Li, "Mercury: A Fast and Energy-Efficient Multi-level Cell based Phase Change Memory System," in *2011 IEEE 17TH INTERNATIONAL SYMPOSIUM ON HIGH-PERFORMANCE COMPUTER ARCHITECTURE (HPCA)*, International Symposium on High-Performance Computer Architecture-Proceedings, (10662 LOS VAQUEROS CIRCLE, PO BOX 3014, LOS ALAMITOS, CA 90720-1264 USA), pp. 345–356, IEEE; IEEE Comp Soc Tech Comm Comp Architecture; IEEE Comp Soc; HP Invent; AMD; CAVIUM Networks; IBM Res; Intel; Microsoft Res; VMware; Univ Texas San Antonio (UTSA), Coll Sci (COS), IEEE COMPUTER SOC, 2011. 17th IEEE International Symposium on High-Performance Computer Architecture (HPCA), San Antonio, TX, FEB 12-16, 2011.

[68] T. Nirschl, J. B. Philipp, T. D. Flapp, G. W. Burr, B. Rajendran, M. H. Leeo, A. Schrott, M. Yang, M. Breitwisch, C. F. Chen, E. Joseph, M. Lamorey, R. Cheek, S. H. Chen, S. Zaidi, S. Raoux, Y. C. Chen, Y. Zhu, R. Bergmann, H. L. Lung, and C. Lam, "Write strategies for 2 and 4-bit multi-level phase-change memory," in *2007 IEEE INTERNATIONAL ELECTRON DEVICES MEETING, VOLS 1 AND 2*, International Electron Devices Meeting, (345 E 47TH ST, NEW YORK, NY 10017 USA), pp. 461–464, IEEE, IEEE, 2007. IEEE International Electron Devices Meeting, Washington, DC, DEC 10-12, 2007.

[69] L. Jiang, B. Zhao, Y. Zhang, J. Yang, and B. R. Childers, "Improving Write Operations in MLC Phase Change Memory," in *2012 IEEE 18TH INTERNATIONAL SYMPOSIUM ON HIGH PERFORMANCE COMPUTER ARCHITECTURE (HPCA)*, International Symposium on High-Performance Computer Architecture-Proceedings, (345 E 47TH ST, NEW YORK, NY 10017 USA), pp. 201–210, IEEE; IEEE Comp Soc Tech Comm Comp Architecture; IEEE Comp Soc; IBM Res; Facebook, IEEE, 2012. 18th IEEE Inter-

national Symposium on High-Performance Computer Architecture (HPCA), New Orleans, LA, FEB 25-29, 2012.

[70] G. F. Close, U. Frey, J. Morrish, R. Jordan, S. Lewis, T. Maffitt, M. Breitwisch, C. Hagleitner, C. Lam, and E. Eleftheriou, "A 512mb phase-change memory (pcm) in 90nm cmos achieving 2b/cell," in *VLSI Circuits (VLSIC), 2011 Symposium on*, pp. 202–203, 2011.

[71] M. K. Qureshi, M. M. Franceschini, and L. A. Lastras-Montano, "Improving Read Performance of Phase Change Memories via Write Cancellation and Write Pausing," in *HPCA-16 2010: SIXTEENTH INTERNATIONAL SYMPOSIUM ON HIGH-PERFORMANCE COMPUTER ARCHITECTURE, PROCEEDINGS*, International Symposium on High-Performance Computer Architecture-Proceedings, (10662 LOS VAQUEROS CIRCLE, PO BOX 3014, LOS ALAMITOS, CA 90720-1264 USA), pp. 163–173, IEEE Comp Soc, Techn Comm Comp Architecture; Natl Sci Fdn; Govt India, Dept Sci & Technol; Govt India, Dept Informat Technol; Govt India, Council Sci & Ind Res; Govt India, Dept Defense & Res Org; Intel; Samsung; IBM; HP; Google; Netapp; AMD; TCS, IEEE COMPUTER SOC, 2010. 16th International Symposium on High-Performance Computer Architecture, Indian Inst Sci, Bangalore, INDIA, JAN 09-14, 2010.

[72] R. Annunziata, P. Zuliani, M. Borghi, G. De Sandre, L. Scotti, C. Prelini, M. Tosi, I. Tortorelli, and F. Pellizzer, "Phase Change Memory Technology for Embedded Non Volatile Memory Applications for 90nm and Beyond," in *2009 IEEE INTERNATIONAL ELECTRON DEVICES MEETING*, International Electron Devices Meeting, (345 E 47TH ST, NEW YORK, NY 10017 USA), pp. 87–90, IEEE Elect Devices Soc, IEEE, 2009. IEEE International Electron Devices Meeting (IEDM 2009), Baltimore, MD, DEC 07-09, 2009.

[73] "International Technology Roadmap for Semiconductors(ITRS), Chapter: Front End Processes (FEP)," 2011. http://www.itrs.net/Links/2011ITRS/2011Chapters/2011FEP.pdf.

[74] "International Technology Roadmap for Semiconductors(ITRS), Table: Front End Processes (FEP)," 2012. http://www.itrs.net/Links/2012ITRS/2012Tables/FEP_2012Tables.xlsx.

[75] J. Liang, R. G. D. Jeyasingh, H.-Y. Chen, and H. S. P. Wong, "An Ultra-Low Reset Current Cross-Point Phase Change Memory With Carbon Nanotube Electrodes," *IEEE TRANSACTIONS ON ELECTRON DEVICES*, vol. 59, pp. 1155–1163, APR 2012.

[76] R. Jeyasingh, J. Liang, M. A. Caldwell, D. Kuzum, and H. S. P. Wong, "Phase Change Memory: Scaling and Applications," in *2012 IEEE CUSTOM INTEGRATED CIRCUITS CONFERENCE (CICC)*, IEEE Custom Integrated Circuits Conference, (345 E 47TH ST, NEW YORK, NY 10017 USA), Elect Devices Soc (ED); Solid-State Circuits Soc (SSC); AMD; Analog Devices; Catalyst Fdn; Intel Corp; NXP; Rambus; Designers Guide Consulting, IEEE, 2012. 34th Annual IEEE Custom Integrated Circuits Conference (CICC), San Jose, CA, SEP 09-12, 2012.

[77] Z. Tőkei and E. Scheer, *Nanoelectronics and Information Technology*, ch. 33, pp. 767–775. John Wiley & Sons, 2012.

[78] "International Technology Roadmap for Semiconductors(ITRS), Chapter: Interconnects(INTC)," 2011. http://www.itrs.net/Links/2011ITRS/2011Chapters/2011Interconnect.pdf.

[79] "International Technology Roadmap for Semiconductors(ITRS), Tables: Interconnects(INTC)," 2012. http://www.itrs.net/Links/2012ITRS/2012Tables/Interconnect_2012Tables.xlsx.

[80] "International Technology Roadmap for Semiconductors(ITRS) Interconnects," 2000. http://www.itrs.net/Links/2000UpdateFinal/Interconnect2000final.pdf.

[81] "International Technology Roadmap for Semiconductors(ITRS) Interconnects," 2003. http://www.itrs.net/Links/2003ITRS/Interconnect2003.pdf.

[82] "International Technology Roadmap for Semiconductors(ITRS) Interconnects," 2007. http://www.itrs.net/Links/2007ITRS/2007_Chapters/2007_Interconnect.pdf.

[83] T. Ohsawa, K. Kai, and K. Murakami, "Optimizing the dram refresh count for merged dram/logic lsis," in *Proceedings of the 1998 international symposium on Low power electronics and design*, pp. 82–87, ACM, 1998.

[84] H. Iwai, "Roadmap for 22 nm and beyond," *MICROELECTRONIC ENGINEERING*, vol. 86, pp. 1520–1528, JUL-SEP 2009. 16th Biennial Conference on Insulating Films on Semiconductors, Cambridge Univ, Clare Coll, Cambridge, ENGLAND, JUN 28-JUL 07, 2009.

[85] T. Maffitt, J. DeBrosse, J. Gabric, E. Gow, M. Lamorey, J. Parenteau, D. Willmott, M. Wood, and W. Gallagher, "Design considerations for MRAM," *IBM JOURNAL OF RESEARCH AND DEVELOPMENT*, vol. 50, pp. 25–39, JAN 2006.

[86] S. Wolf, A. Chtchelkanova, and D. Treger, "Spintronics - A retrospective and perspective," *IBM JOURNAL OF RESEARCH AND DEVELOPMENT*, vol. 50, pp. 101–110, JAN 2006.

[87] W. Zhao, S. Chaudhuri, C. Accoto, J.-O. Klein, C. Chappert, and P. Mazoyer, "Cross-Point Architecture for Spin-Transfer Torque Magnetic Random Access Memory," *IEEE TRANSACTIONS ON NANOTECHNOLOGY*, vol. 11, pp. 907–917, SEP 2012.

[88] C. Xu, D. Niu, X. Zhu, S. H. Kang, M. Nowak, and Y. Xie, "Device-Architecture Co-Optimization of STT-RAM Based Memory for Low Power Embedded Systems," in *2011 IEEE/ACM INTERNATIONAL CONFERENCE ON COMPUTER-AIDED DESIGN (ICCAD)*, (345 E 47TH ST, NEW YORK, NY 10017 USA), pp. 463–470, IEEE; IEEE Council Elect Design Automat (CEDA); Special Interest Grp Design Automat (SICDA); ACM, IEEE, 2011. IEEE/ACM International Conference on Computer-Aided Design (ICCAD), San Jose, CA, NOV 07-10, 2011.

[89] A. Pirovano, A. Lacaita, F. Pellizzer, S. Kostylev, A. Benvenuti, and R. Bez, "Low-field amorphous state resistance and threshold voltage drift in chalcogenide materials," *IEEE TRANSACTIONS ON ELECTRON DEVICES*, vol. 51, pp. 714–719, MAY 2004.

[90] U. Russo, D. Ielmini, A. Redaelli, and A. L. Lacaita, "Modeling of programming and read performance in phase-change memories - Part I: Cell optimization and scaling," *IEEE TRANSACTIONS ON ELECTRON DEVICES*, vol. 55, pp. 506–514, FEB 2008.

[91] A. Gyanathan and Y.-C. Yeo, "Multi-level phase change memory devices with Ge2Sb2Te5 layers separated by a thermal insulating Ta2O5 barrier layer," *JOURNAL OF APPLIED PHYSICS*, vol. 110, DEC 15 2011.

[92] G. De Sandre, L. Bettini, A. Pirola, L. Marmonier, M. Pasotti, M. Borghi, P. Mattavelli, P. Zuliani, L. Scotti, G. Mastracchio, F. Bedeschi, R. Gastaldi, and R. Bez, "A 4 Mb LV MOS-Selected Embedded Phase Change Memory in 90 nm Standard CMOS Technology," *IEEE JOURNAL OF SOLID-STATE CIRCUITS*, vol. 46, pp. 52–63, JAN 2011. International Solid-State Circuits Conference (ISSCC), San Francisco, CA, FEB 07-11, 2010.

[93] N. Takaura, A. Terao, K. Kurotsuchi, T. Yamauchi, O. Tonomura, Y. Hanaoka, R. Takemura, K. Osada, T. Kawahara, and H. Matsuoka, "A GeSbTe phase-change memory cell featuring a tungsten heater electrode for low-power, highly stable, and short-read-cycle operations," in *2003 IEEE INTERNATIONAL ELECTRON DEVICES MEETING, TECHNICAL DI-*

GEST, (345 E 47TH ST, NEW YORK, NY 10017 USA), pp. 897–900, IEEE, Elect Devices Soc, IEEE, 2003. IEEE International Electron Devices Meeting, WASHINGTON, D.C., DEC 08-10, 2003.

[94] K. Johguchi, T. Shintani, T. Morikawa, K. Yoshioka, and K. Takeuchi, “x10 Fast write, 80% energy saving temperature controlling set method for multi-level cell phase change memories to solve the scaling blockade,” *SOLID-STATE ELECTRONICS*, vol. 81, pp. 78–85, MAR 2013.

[95] Y. Yin, T. Noguchi, and S. Hosaka, “Possibility of Freely Achievable Multilevel Storage of Phase-Change Memory by Staircase-Shaped Pulse Programming,” *JAPANESE JOURNAL OF APPLIED PHYSICS*, vol. 50, OCT 2011.

[96] C. Resta, M. Ferraro, F. Bedeschi, and A. Cabrini, “Programming a multi-level phase change memory cell,” Aug. 31 2010. US Patent 7,787,291.

[97] F. Bedeschi, R. Fackenthal, C. Resta, E. M. Donze, M. Jagasivamani, E. C. Buda, F. Pellizzer, D. W. Chow, A. Cabrini, G. M. A. Calvi, R. Faravelli, A. Fantini, G. Torelli, D. Mills, R. Gastaldi, and G. Casagrande, “A Bipolar-Selected Phase Change Memory Featuring Multi-Level Cell Storage,” *IEEE JOURNAL OF SOLID-STATE CIRCUITS*, vol. 44, pp. 217–227, JAN 2009. IEEE International Solid-State Circuits Conference (ISSCC), San Francisco, CA, FEB 03-07, 2008.

[98] S. Braga, A. Sanasi, A. Cabrini, and G. Torelli, “Voltage-Driven Partial-RESET Multilevel Programming in Phase-Change Memories,” *IEEE TRANSACTIONS ON ELECTRON DEVICES*, vol. 57, pp. 2556–2563, OCT 2010.

[99] N. Papandreou, H. Pozidis, A. Pantazi, A. Sebastian, M. Breitwisch, C. Lam, and E. Eleftheriou, “Programming Algorithms for Multilevel Phase-Change Memory,” in *2011 IEEE INTERNATIONAL SYMPOSIUM ON CIRCUITS AND SYSTEMS (ISCAS)*, IEEE International Symposium on Circuits and Systems, (345 E 47TH ST, NEW YORK, NY 10017 USA), pp. 329–332, IEEE, IEEE, 2011. IEEE International Symposium on Circuits and Systems (ISCAS), Rio de Janeiro, BRAZIL, MAY 15-18, 2011.

[100] A. Cabrini, S. Braga, A. Manetto, and G. Torelli, “Voltage-Driven Multi-level Programming in Phase Change Memories,” in *2009 IEEE INTERNATIONAL WORKSHOP ON MEMORY TECHNOLOGY, DESIGN, AND TESTING, PROCEEDINGS*, (10662 LOS VAQUEROS CIRCLE, PO BOX 3014, LOS ALAMITOS, CA 90720-1264 USA), pp. 3–6, IEEE, IEEE COM-

PUTER SOC, 2009. IEEE International Workshop on Memory Technology, Design and Testing, Hsinchu, TAIWAN, AUG 31-SEP 02, 2009.

[101] M. K. Qureshi, M. M. Franceschini, L. A. Lastras-Montano, and J. P. Karidis, "Morphable Memory System: A Robust Architecture for Exploiting Multi-Level Phase Change Memories," in *ISCA 2010: THE 37TH ANNUAL INTERNATIONAL SYMPOSIUM ON COMPUTER ARCHITECTURE*, Conference Proceedings Annual International Symposium on Computer Architecture, (1515 BROADWAY, NEW YORK, NY 10036-9998 USA), pp. 153–162, ACM SIGARCH; IEEE Comp Soc; IEEE TCCA, ASSOC COMPUTING MACHINERY, 2010. 37th International Symposium on Computer Architecture, St Malo, FRANCE, JUN 19-23, 2010.

[102] T. Skotnicki, F. Boeuf, M. Müller, A. Pouydebasque, C. Fenouillet-Béranger, R. Cerutti, S. Harrison, S. Monfray, B. Dumont, and F. Payet, *A users guide to MASTAR 4*. http://www.itrs.net/Links/2011ITRS/MASTAR2011/instructions.pdf.

[103] A. Agarwal and J. Lang, *Foundations of analog and digital electronic circuits*, ch. 15, pp. 837–902. Morgan Kaufmann, 2005.

[104] H. R. Huff and D. D. C. Gilmer, *High dielectric constant materials: VLSI MOSFET Applications*, vol. 16. Springer, 2005.

[105] "International Technology Roadmap for Semiconductors(ITRS), Tables: Process Integration, Devices and Structures(PIDS)," 2008. http://www.itrs.net/Links/2008ITRS/Update/2008Tables_FOCUS_A.xls.

[106] "International Technology Roadmap for Semiconductors(ITRS), Tables: Process Integration, Devices and Structures(PIDS)," 2000. http://www.itrs.net/Links/2000UpdateFinal/ProcessInt2000final.pdf.

[107] "International Technology Roadmap for Semiconductors(ITRS), Tables: Process Integration, Devices and Structures(PIDS)," 2004. http://www.itrs.net/Links/2004Update/2004_03_PIDS.pdf.

[108] *EFM32LG Data sheet*, (rev 0.90 - 10/11) ed. http://cdn.energymicro.com/dl/devices/pdf/d0120_efm32lg940_datasheet.pdf.

[109] *EFM32TG110 Data sheet*, (rev 0.91 - 02/11) ed. http://cdn.energymicro.com/dl/devices/pdf/d0013_efm32tg110_datasheet.pdf.

[110] E. SEEVINCK, P. VANBEERS, and H. ONTROP, "CURRENT-MODE TECHNIQUES FOR HIGH-SPEED VLSI CIRCUITS WITH APPLICA-

TION TO CURRENT SENSE AMPLIFIER FOR CMOS SRAMS," *IEEE JOURNAL OF SOLID-STATE CIRCUITS*, vol. 26, pp. 525–536, APR 1991.

[111] S. MILLER, R. NASBY, J. SCHWANK, M. RODGERS, and P. DRESSENDORFER, "DEVICE MODELING OF FERROELECTRIC CAPACITORS," *JOURNAL OF APPLIED PHYSICS*, vol. 68, pp. 6463–6471, DEC 15 1990.

[112] R. MOAZZAMI, C. HU, and W. SHEPHERD, "ELECTRICAL CHARACTERISTICS OF FERROELECTRIC PZT THIN-FILMS FOR DRAM APPLICATIONS," *IEEE TRANSACTIONS ON ELECTRON DEVICES*, vol. 39, pp. 2044–2049, SEP 1992.

[113] "International Technology Roadmap for Semiconductors(ITRS), Tables: Process Integration, Devices and Structures(PIDS)," 2007. http://www.itrs.net/Links/2007ITRS/2007_Chapters/2007_PIDS.pdf.

[114] "International Technology Roadmap for Semiconductors(ITRS), Tables: Process Integration, Devices and Structures(PIDS)," 2009. http://www.itrs.net/Links/2009ITRS/2009Chapters_2009Tables/2009Tables_FOCUS_C_ITRS.xls.

[115] "Everspin debuts first Spin-Torque MRAM for high performance storage systems," November 2012. http://www.everspin.com/technology.php?qtype=3.

[116] *MB85R256F Data sheet*. http://www.fujitsu.com/downloads/MICRO/fme/fram/datasheet-MB85R256F.pdf.

[117] *FM22LD16 Data sheet*. http://www.ramtron.com/files/datasheets/FM22LD16_ds.pdf.

[118] *MR256A08B Data sheet*. http://www.everspin.com/PDF/EST_MR256A08B_prod.pdf.

[119] *MR2A16A Data sheet*. http://www.everspin.com/PDF/EST_MR2A16A_prod.pdf.

[120] *NP8P128A13TSM60E Data sheet*. http://www.micron.com/~/media/Documents/Products/Data%20Sheet/PCM/p8p_parallel_pcm_ds.pdf.

[121] *IS61LV25616AL Data sheet*. http://www.issi.com/WW/pdf/61LV25616AL.pdf.

[122] *CY62146EV30 Data sheet*. http://www.cypress.com/?docID=43021.

[123] *EFM32GG980 Data sheet.* http://cdn.energymicro.com/dl/devices/pdf/d0045_efm32gg980_datasheet.pdf.

[124] I. Helland, May 2013. Private communication.

[125] *Hardware Design Considerations – AN0002 Application Note.* http://cdn.energymicro.com/dl/an/pdf/an0002_efm32_hardware_design_considerations.pdf.

[126] *Oscillator Design Considerations – AN0016 Application Note.* http://cdn.energymicro.com/dl/an/pdf/an0016_efm32_oscillator_design_considerations.pdf.

[127] *EFM32GG Reference Manual.* http://cdn.energymicro.com/dl/devices/pdf/d0053_efm32gg_reference_manual.pdf.

[128] *External Bus Interface – AN0034 Application Note.* http://cdn.energymicro.com/dl/an/pdf/an0034_efm32_ebi.pdf.

[129] "Technical Note – Software Device Drivers for Micron P8P Parallel Phase Change Memory." http://www.micron.com/~/media/Documents/Products/Technical%20Note/Phase%20Change%20Memory/tn1306_P8P_software_device_drivers.pdf.

[130] T. Ohsawa, H. Koike, S. Miura, H. Honjo, K. Tokutome, S. Ikeda, T. Hanyu, H. Ohno, and T. Endoh, "1mb 4t-2mtj nonvolatile stt-ram for embedded memories using 32b fine-grained power gating technique with 1.0 ns/200ps wake-up/power-off times," in *VLSI Circuits (VLSIC), 2012 Symposium on*, pp. 46–47, IEEE, 2012.

[131] T. Kawahara, "Challenges toward gigabit-scale spin-transfer torque random access memory and beyond for normally off, green information technology infrastructure (invited)," *JOURNAL OF APPLIED PHYSICS*, vol. 109, APR 1 2011.

[132] G. Sun, D. Niu, J. Ouyang, and Y. Xie, "A Frequent-Value Based PRAM Memory Architecture," in *2011 16TH ASIA AND SOUTH PACIFIC DESIGN AUTOMATION CONFERENCE (ASP-DAC)*, Asia and South Pacific Design Automation Conference Proceedings, (345 E 47TH ST, NEW YORK, NY 10017 USA), IEEE, 2011. 16th Asia and South Pacific Design Automation Conference (ASP-DAC), Yokohama, JAPAN, JAN 25-28, 2011.

A Appendix: Matlab Scripts for Theoretical Estimates

In this appendix the Matlab scripts developed to calculate the power consumption and generate the plots are included.

A.1 Main Script

This is the main script which in turn calls the other scripts and then produces the plots.

```
clc
clear all
close all
% Input parameters:
number_of_bytes=32*1024; % 32 KB
f=32*1e6;
word_size=16;

percentage_write=0.4;
percentage_read=1-percentage_write;

N_WL_squareroot=true;
%If true the number of wordlines is the the squareroot of the
    number of
%cells needed, if false the number of cells is given by N_WL
N_WL=[512 512 512];

%Data is stored in arrays in following format:
% Variable_name=[2012_data 2017_data iso_F_data];


% if true all the plots will be saved as pdfs
save=true;

% use new interconnect model.
new = true;

%Scaling for text on pictures:
d=2;
scale=1.5;

run DRAM
run SRAM
```

```
33 run FeRAM
34 run Conv_MRAM
35 run STT_MRAM
36 run PCRAM
37
38 groups={'DRAM','SRAM','FeRAM','MRAM','STT-MRAM','PCRAM'};
39 warning('off','MATLAB:Axes:NegativeDataInLogAxis')
40 density=[DRAM_density;SRAM_density; FeRAM_density;...
41     Conv_MRAM_density;STT_MRAM_density;PCRAM_density]*1e-12;
42
43 figure;
44 bar(density(:,1:2))
45 barnumber(density(:,1:2),length(groups));
46 set(gca,'XTickLabel',groups)
47 legend('2012','2017','Location','NorthWest')
48 ylabel('Number of bits per  \mum^2')
49 title('Bit density')
50 logbar()
51 if (save)
52     saveas(gcf,'area','pdf')
53 end
54
55 P_write=[DRAM_P_write; SRAM_P_write; FeRAM_P_write;...
56     Conv_MRAM_P_write;STT_MRAM_P_write; PCRAM_P_write]*1e6;
57
58 figure;
59 bar(P_write(:,1:2),'BaseValue',0.01)
60 barnumber(P_write(:,1:2),length(groups));
61 set(gca,'XTickLabel',groups)
62 legend('2012','2017','Location','NorthWest')
63 ylabel('P_{write} [\muW]')
64 title('P_{write}')
65 logbar()
66 ylim([1 100000])
67 if (save)
68 saveas(gcf,'writepower','pdf')
69 end
70
71 P_read=[DRAM_P_read; SRAM_P_read; FeRAM_P_read; ...
72     Conv_MRAM_P_read;STT_MRAM_P_read; PCRAM_P_read]*1e6;
73
74 figure;
75 bar(P_read(:,1:2),'BaseValue',0.01)
76 barnumber(P_read(:,1:2),length(groups));
77 set(gca,'XTickLabel',groups)
78 legend('2012','2017','Location','NorthWest')
79 ylabel('P_{read} [\muW]')
80 title('P_{read}')
81 logbar()
```

```
82 ylim([0.1 1000])
83 if( save)
84 saveas(gcf,'readpower','pdf')
85 end
86
87 E_write=[DRAM_E_write; SRAM_E_write; FeRAM_E_write;...
88     Conv_MRAM_E_write; STT_MRAM_E_write; PCRAM_E_write]*1e15;
89
90 figure;
91 bar(E_write(:,1:2),'BaseValue',0.01)
92
93 barnumber(E_write(:,1:2),length(groups));
94 set(gca,'XTickLabel',groups)
95 legend('2012','2017','Location','NorthWest')
96 ylabel('E_{cell-write/bit} [fJ]')
97 title('E_{cell-write/bit}')
98 ylim([0.01 120000])
99 logbar()
100 if (save)
101 saveas(gcf,'cell_write_energy','pdf')
102 end
103
104 E_read=[DRAM_E_write;SRAM_E_read; FeRAM_E_write;...
105     Conv_MRAM_E_read;STT_MRAM_E_read; PCRAM_E_read]*1e15;
106
107 figure;
108 bar(E_read(:,1:2),'BaseValue',0.01)
109 barnumber(E_read(:,1:2),length(groups));
110 set(gca,'XTickLabel',groups)
111 legend('2012','2017','Location','NorthWest')
112 ylabel('E_{cell-read/bit} [fJ]')
113 title('E_{cell-read/bit}')
114 logbar()
115 ylim([0.1 1000])
116 if (save)
117 saveas(gcf,'cell_read_energy','pdf')
118 end
119
120 S=[DRAM_S;SRAM_S; FeRAM_S;...
121     Conv_MRAM_S;STT_MRAM_S; PCRAM_S];
122
123 figure;
124 bar(S(:,1:2),'BaseValue',0.5e-1)
125 %barnumber(S(:,1:2),length(groups));
126 set(gca,'XTickLabel',groups)
127 legend('2012','2017','Location','NorthWest')
128 ylabel('S')
129 title('Relative signal strength')
130 %logbar()
```

```
131 ylim([0.5e-1 1])
132 if (save)
133 saveas(gcf,'Signal_strength','pdf')
134 end
135 P_refresh=[DRAM_P_refresh;SRAM_P_retention;FeRAM_P_refresh;...
136     Conv_MRAM_P_refresh;STT_MRAM_P_refresh;PCRAM_P_refresh];
137
138 E_T=[DRAM_E_T;SRAM_E_T;FeRAM_E_T;...
139     Conv_MRAM_E_T;STT_MRAM_E_T;PCRAM_E_T_read]*1e15;
140 figure;
141 bar(E_T(:,1:2),'BaseValue',1e-2)
142 barnumber(E_T(:,1:2),length(groups));
143 set(gca,'XTickLabel',groups)
144 legend('2012','2017','Location','NorthWest')
145 ylabel('E_T [fJ]')
146 title('Energy required to switch the access transistors')
147 logbar()
148 ylim([1e0 1e3])
149 if (save)
150 saveas(gcf,'E_T','pdf')
151 end
152
153
154 %%%%%%%%%%%%%%%%%%%%%%%%%%%%
155 min_X=-6;
156 t=logspace(min_X,0,1000);
157
158 DRAM_P_2012=DRAM_P_refresh(1,1)+t.*...
159     (percentage_write.*DRAM_P_write(1,1)...
160     +percentage_read.*DRAM_P_read(1,1));
161 SRAM_P_2012=SRAM_P_retention(1,1)+t.*...
162     (percentage_write.*SRAM_P_write(1,1)...
163     +percentage_read.*SRAM_P_read(1,1));
164 FeRAM_P_2012=FeRAM_P_refresh(1,1)+t.*...
165     (percentage_write.*FeRAM_P_write(1,1)+...
166     percentage_read.*FeRAM_P_read(1,1));
167 Conv_MRAM_P_2012=Conv_MRAM_P_refresh(1,1)+t.*...
168     (percentage_write.*Conv_MRAM_P_write(1,1)...
169     +percentage_read.*Conv_MRAM_P_read(1,1));
170 STT_MRAM_P_2012=STT_MRAM_P_refresh(1,1)+t.*...
171     (percentage_write.*STT_MRAM_P_write(1,1)...
172     +percentage_read.*STT_MRAM_P_read(1,1));
173 PCRAM_P_2012=PCRAM_P_refresh(1,1)+t.*...
174     (percentage_write.*PCRAM_P_write(1,1)...
175     +percentage_read.*PCRAM_P_read(1,1));
176 figure;
177 loglog(t,DRAM_P_2012,'b')
178 hold on
179 loglog(t,SRAM_P_2012,'c')
```

```
180 loglog(t,FeRAM_P_2012,'g')
181 loglog(t,Conv_MRAM_P_2012,'Color',[1,0.5,0])
182 loglog(t,STT_MRAM_P_2012,'r')
183 loglog(t,PCRAM_P_2012,'m')
184 legend(groups,'Location','NorthWest')
185 title('Power vs Duty cycle 2012')
186 ylabel('P [W]')
187 xlim([min_X 1e0])
188 ylim([1e-10 1e-1])
189 xlabel('Duty cycle ')
190 [xout,yout] = intersections(t,DRAM_P_2012,t,FeRAM_P_2012,1);
191 loglog(xout,yout,'g.','markersize',10)
192 text(xout,yout/scale,char(vpa(xout,d)),'fontsize',12,'
        horizontalAlignment','left','color','g')
193
194 [xout,yout] = intersections(t,SRAM_P_2012,t,FeRAM_P_2012,1);
195 loglog(xout,yout,'g.','markersize',10)
196 text(xout,yout/scale,char(vpa(xout,d)),'fontsize',12,'fontsize'
        ,12,'horizontalAlignment','left','Color','g')
197
198
199 [xout,yout] = intersections(t,DRAM_P_2012,t,Conv_MRAM_P_2012,1);
200 loglog(xout,yout,'.','Color',[1,0.5,0],'markersize',10)
201 text(xout,yout/scale,char(vpa(xout,d)),'fontsize',12,'
        horizontalAlignment','left','Color',[1,0.5,0])
202
203 [xout,yout] = intersections(t,SRAM_P_2012,t,Conv_MRAM_P_2012,1);
204 loglog(xout,yout,'.','Color',[1,0.5,0],'markersize',10)
205 text(xout,yout*scale,char(vpa(xout,d)),'fontsize',12,'
        horizontalAlignment','right','color',[1,0.5,0])
206
207
208 [xout,yout] = intersections(t,DRAM_P_2012,t,STT_MRAM_P_2012,1);
209 plot(xout,yout,'r.','markersize',10)
210 text(xout,yout/scale,char(vpa(xout,d)),'fontsize',12,'
        horizontalAlignment','right','color','r')
211
212 [xout,yout] = intersections(t,SRAM_P_2012,t,STT_MRAM_P_2012,1);
213 plot(xout,yout,'r.','markersize',10)
214 text(xout,yout/scale,char(vpa(xout,d)),'fontsize',12,'
        horizontalAlignment','right','color','r')
215 %
216 %
217 [xout,yout] = intersections(t,DRAM_P_2012,t,PCRAM_P_2012,1);
218 plot(xout,yout,'m.','markersize',10)
219 text(xout,yout*(scale*1),char(vpa(xout,d)),'fontsize',12,'
        horizontalAlignment','right','color','m')
220
221 [xout,yout] = intersections(t,SRAM_P_2012,t,PCRAM_P_2012,1);
```

```
222 plot(xout,yout,'m.','markersize',10)
223 text(xout,yout*scale,char(vpa(xout,d)),'fontsize',12,'
        horizontalAlignment','right','color','m')
224 if (save)
225 saveas(gcf,'PvsDutycycle2012','pdf')
226 end
227
228 DRAM_P_2017=DRAM_P_refresh(1,2)+t.*...
229      (percentage_write.*DRAM_P_write(1,2)...
230      +percentage_read.*DRAM_P_read(1,2));
231 SRAM_P_2017=SRAM_P_retention(1,2)+t.*...
232      (percentage_write.*SRAM_P_write(1,2)...
233      +percentage_read.*SRAM_P_read(1,2));
234 FeRAM_P_2017=FeRAM_P_refresh(1,2)+t.*...
235      (percentage_write.*FeRAM_P_write(1,2)...
236      +percentage_read.*FeRAM_P_read(1,2));
237 Conv_MRAM_P_2017=Conv_MRAM_P_refresh(1,2)+t.*...
238      (percentage_write.*Conv_MRAM_P_write(1,2)...
239      +percentage_read.*Conv_MRAM_P_read(1,2));
240 STT_MRAM_P_2017=STT_MRAM_P_refresh(1,2)+t.*...
241      (percentage_write.*STT_MRAM_P_write(1,2)...
242      +percentage_read.*STT_MRAM_P_read(1,2));
243 PCRAM_P_2017=PCRAM_P_refresh(1,2)+t.*...
244      (percentage_write.*PCRAM_P_write(1,2)...
245      +percentage_read.*PCRAM_P_read(1,2));
246 figure;
247 loglog(t,DRAM_P_2017,'b')
248 hold on
249 loglog(t,SRAM_P_2017,'c')
250 loglog(t,FeRAM_P_2017,'g')
251 loglog(t,Conv_MRAM_P_2017,'Color',[1,0.5,0])
252 loglog(t,STT_MRAM_P_2017,'r')
253 loglog(t,PCRAM_P_2017,'m')
254 legend(groups,'Location','NorthWest')
255 title('Power vs Duty cycle 2017')
256 ylabel('P [W]')
257 xlim([min_X 1e0])
258 ylim([1e-10 1e-1])
259 xlabel('Duty cycle ')
260 [xout,yout] = intersections(t,DRAM_P_2017,t,FeRAM_P_2017,1);
261 loglog(xout,yout,'g.','markersize',10)
262 text(xout,yout*scale,char(vpa(xout,d)),'fontsize',12,'
        horizontalAlignment','right','color','g')
263
264 [xout,yout] = intersections(t,SRAM_P_2017,t,FeRAM_P_2017,1);
265 loglog(xout,yout,'g.','markersize',10)
266 text(xout,yout*scale,char(vpa(xout,d)),'fontsize',12,'
        horizontalAlignment','right','Color','g')
267
```

```
268
269 [xout,yout] = intersections(t,DRAM_P_2017,t,Conv_MRAM_P_2017,1);
270 loglog(xout,yout,'.','Color',[1,0.5,0],'markersize',10)
271 text(xout,yout/scale,char(vpa(xout,d)),'fontsize',12,'
        horizontalAlignment','left','Color',[1,0.5,0])
272
273 [xout,yout] = intersections(t,SRAM_P_2017,t,Conv_MRAM_P_2017,1);
274 loglog(xout,yout,'.','Color',[1,0.5,0],'markersize',10)
275 text(xout,yout*scale,char(vpa(xout,d)),'fontsize',12,'
        horizontalAlignment','right','color',[1,0.5,0])
276
277
278 [xout,yout] = intersections(t,DRAM_P_2017,t,STT_MRAM_P_2017,1);
279 plot(xout,yout,'r.','markersize',10)
280 text(xout,yout/scale,char(vpa(xout,d)),'fontsize',12,'
        horizontalAlignment','left','color','r')
281
282 [xout,yout] = intersections(t,SRAM_P_2017,t,STT_MRAM_P_2017,1);
283 plot(xout,yout,'r.','markersize',10)
284 text(xout,yout/scale,char(vpa(xout,d)),'fontsize',12,'
        horizontalAlignment','left','color','r')
285 %
286 %
287 [xout,yout] = intersections(t,DRAM_P_2017,t,PCRAM_P_2017,1);
288 plot(xout,yout,'m.','markersize',10)
289 text(xout,yout/scale,char(vpa(xout,d)),'fontsize',12,'
        horizontalAlignment','right','color','m')
290
291 [xout,yout] = intersections(t,SRAM_P_2017,t,PCRAM_P_2017,1);
292 plot(xout,yout,'m.','markersize',10)
293 text(xout,yout/scale,char(vpa(xout,d)),'fontsize',12,'
        horizontalAlignment','left','color','m')
294
295 if( save)
296 saveas(gcf,'PvsDutycycle2017','pdf')
297 end
298 %Breakdown
299 P_write_breakdown_2012=[DRAM_P_write_breakdown(:,1)';...
300     SRAM_P_write_breakdown(:,1)';...
301     FeRAM_P_write_breakdown(:,1)';...
302     Conv_MRAM_P_write_breakdown(:,1)';...
303     STT_MRAM_P_write_breakdown(:,1)';...
304     PCRAM_P_write_breakdown(:,1)']*1e6;
305
306
307 figure;
308 bar(P_write_breakdown_2012,'BaseValue',0.001)
309 set(gca,'XTickLabel',groups)
310 legend('Charging word lines','Charging bit lines','Switching cells
```

```
      ',...
311     'Switching access transistors','Location','NorthWest')
312 ylabel('P [\muW]')
313 title('Different contributions to P_{write} in 2012')
314 logbar()
315 ylim([0.001 100000])
316 if (save)
317 saveas(gcf,'breakdown_of_write_power_2012','pdf')
318 end
319
320 %Breakdown
321 P_write_breakdown_2017=[DRAM_P_write_breakdown(:,2)';...
322     SRAM_P_write_breakdown(:,2)';...
323     FeRAM_P_write_breakdown(:,2)';...
324     Conv_MRAM_P_write_breakdown(:,2)';...
325     STT_MRAM_P_write_breakdown(:,2)';...
326     PCRAM_P_write_breakdown(:,2)']*1e6;
327
328
329
330 figure;
331 bar(P_write_breakdown_2017,'BaseValue',0.001)
332 set(gca,'XTickLabel',groups)
333 legend('Charging word lines','Charging bit lines','Switching cells
      ',...
334     'Switching access transistors','Location','NorthWest')
335 ylabel('P [\muW]')
336 title('Different contributions to P_{write} in 2017')
337 logbar()
338 ylim([0.001 100000])
339 if (save)
340 saveas(gcf,'breakdown_of_write_power_2017','pdf')
341 end
342
343
344 % Read breakdown
345 P_read_breakdown_2012=[DRAM_P_read_breakdown(:,1)';...
346     SRAM_P_read_breakdown(:,1)';...
347     FeRAM_P_read_breakdown(:,1)';...
348     Conv_MRAM_P_read_breakdown(:,1)';...
349     STT_MRAM_P_read_breakdown(:,1)';...
350     PCRAM_P_read_breakdown(:,1)']*1e6;
351
352
353 figure;
354 bar(P_read_breakdown_2012,'BaseValue',0.001)
355 set(gca,'XTickLabel',groups)
356 legend('Charging word lines','Charging bit lines','Cell read power
      ',...
```

```
357     'Switching access transistors','Location','NorthWest')
358 ylabel('P [\muW]')
359 title('Different contributions to P_{read} in 2012')
360 logbar()
361 ylim([0.01 10000])
362 if (save)
363 saveas(gcf,'breakdown_of_read_power_2012','pdf')
364 end
365
366 % Read breakdown
367 P_read_breakdown_2017=[DRAM_P_read_breakdown(:,2)';...
368     SRAM_P_read_breakdown(:,2)';...
369     FeRAM_P_read_breakdown(:,2)';...
370     Conv_MRAM_P_read_breakdown(:,2)';...
371     STT_MRAM_P_read_breakdown(:,2)';...
372     PCRAM_P_read_breakdown(:,2)']*1e6;
373
374
375 figure;
376 bar(P_read_breakdown_2017,'BaseValue',0.001)
377 set(gca,'XTickLabel',groups)
378 legend('Charging word lines','Charging bit lines','Cell read power
    ',...
379     'Switching access transistors','Location','NorthWest')
380 ylabel('P [\muW]')
381 title('Different contributions to P_{read} in 2017')
382 logbar()
383 ylim([0.01 10000])
384 if (save)
385 saveas(gcf,'breakdown_of_read_power_2017','pdf')
386 end
387
388
389
390 %%%%%%%%%%%%%%%%%%%%%%%%
391 % ISO Feature size
392 figure;
393 bar(density(:,3))
394 barnumber(density(:,3),length(groups));
395 set(gca,'XTickLabel',groups)
396 %legend('65 nm','Location','NorthWest')
397 ylabel('Number of bits per  \mum^2')
398 title('Bit density for F=65 nm')
399 logbar()
400 if (save)
401     saveas(gcf,'area_iso_F','pdf')
402 end
403
404 figure;
```

```
405 bar([P_write(:,3) P_read(:,3)],'BaseValue',0.01)
406 barnumber([P_write(:,3) P_read(:,3)],length(groups));
407 set(gca,'XTickLabel',groups)
408 legend('P_{write}','P_{read}','Location','NorthWest')
409 ylabel('P [\muW]')
410 title('P_{write} and P_{read} for F=65 nm')
411 logbar()
412 ylim([1 100000])
413 if (save)
414 saveas(gcf,'writepower_iso_F','pdf')
415 end
416
417
418 figure;
419 bar([E_write(:,3) E_read(:,3)],'BaseValue',0.01)
420 barnumber([E_write(:,3) E_read(:,3)],length(groups));
421 set(gca,'XTickLabel',groups)
422 legend('E_{cell-write/bit}','E_{cell-read/bit}','Location','
        NorthWest')
423 ylabel('E [fJ]')
424 title('E_{cell-write/bit} and E_{cell-write/bit} for F=65 nm')
425 ylim([0.1 1000000])
426 logbar()
427 if (save)
428 saveas(gcf,'cell_energy_iso_F','pdf')
429 end
430
431
432 P_write_breakdown_iso_F=[DRAM_P_write_breakdown(:,3)';...
433      SRAM_P_write_breakdown(:,3)';...
434      FeRAM_P_write_breakdown(:,3)';...
435      Conv_MRAM_P_write_breakdown(:,3)';...
436      STT_MRAM_P_write_breakdown(:,3)';...
437      PCRAM_P_write_breakdown(:,3)']*1e6;
438
439
440 figure;
441 bar(P_write_breakdown_iso_F,'BaseValue',0.001)
442 set(gca,'XTickLabel',groups)
443 legend('Charging word lines','Charging bit lines','Switching cells
        ',...
444      'Switching access transistors','Location','NorthWest')
445 ylabel('P [\muW]')
446 title('Different contributions to P_{write} for F=65 nm')
447 logbar()
448 ylim([0.001 100000])
449 if (save)
450 saveas(gcf,'breakdown_of_write_power_iso_F','pdf')
451 end
```

```
452
453 P_read_breakdown_iso_F=[DRAM_P_read_breakdown(:,3)';...
454     SRAM_P_read_breakdown(:,3)';...
455     FeRAM_P_read_breakdown(:,3)';...
456     Conv_MRAM_P_read_breakdown(:,3)';...
457     STT_MRAM_P_read_breakdown(:,3)';...
458     PCRAM_P_read_breakdown(:,3)']*1e6;
459
460
461 figure;
462 bar(P_read_breakdown_iso_F,'BaseValue',0.001)
463 set(gca,'XTickLabel',groups)
464 legend('Charging word lines','Charging bit lines','Cell read power
    ',...
465     'Switching access transistors','Location','NorthWest')
466 ylabel('P [\muW]')
467 title('Different contributions to P_{read} for F=65 nm')
468 logbar()
469 ylim([0.001 100000])
470 if (save)
471 saveas(gcf,'breakdown_of_read_power_iso_F','pdf')
472 end
473
474
475 DRAM_P_iso_F=DRAM_P_refresh(1,3)+t.*...
476     (percentage_write.*DRAM_P_write(1,3)...
477     +percentage_read.*DRAM_P_read(1,3));
478 SRAM_P_iso_F=SRAM_P_retention(1,3)+t.*...
479     (percentage_write.*SRAM_P_write(1,3)...
480     +percentage_read.*SRAM_P_read(1,3));
481 FeRAM_P_iso_F=FeRAM_P_refresh(1,3)+t.*...
482     (percentage_write.*FeRAM_P_write(1,3)+...
483     percentage_read.*FeRAM_P_read(1,3));
484 Conv_MRAM_P_iso_F=Conv_MRAM_P_refresh(1,3)+t.*...
485     (percentage_write.*Conv_MRAM_P_write(1,3)...
486     +percentage_read.*Conv_MRAM_P_read(1,3));
487 STT_MRAM_P_iso_F=STT_MRAM_P_refresh(1,3)+t.*...
488     (percentage_write.*STT_MRAM_P_write(1,3)...
489     +percentage_read.*STT_MRAM_P_read(1,3));
490 PCRAM_P_iso_F=PCRAM_P_refresh(1,3)+t.*...
491     (percentage_write.*PCRAM_P_write(1,3)...
492     +percentage_read.*PCRAM_P_read(1,3));
493 figure;
494 loglog(t,DRAM_P_iso_F,'b')
495 hold on
496 loglog(t,SRAM_P_iso_F,'c')
497 loglog(t,FeRAM_P_iso_F,'g')
498 loglog(t,Conv_MRAM_P_iso_F,'Color',[1,0.5,0])
499 loglog(t,STT_MRAM_P_iso_F,'r')
```

```
500 loglog(t,PCRAM_P_iso_F,'m')
501 legend(groups,'Location','NorthWest')
502 title('Power vs Duty cycle for F=65 nm')
503 ylabel('P [W]')
504 ylim([1e-10 1e-1])
505 xlim([min_X 1e0])
506 xlabel('Duty cycle ')
507 [xout,yout] = intersections(t,DRAM_P_iso_F,t,FeRAM_P_iso_F,1);
508 loglog(xout,yout,'g.','markersize',10)
509 text(xout,yout/scale,char(vpa(xout,d)),'fontsize',12,'
        horizontalAlignment','left','color','g')
510
511 [xout,yout] = intersections(t,SRAM_P_iso_F,t,FeRAM_P_iso_F,1);
512 loglog(xout,yout,'g.','markersize',10)
513 text(xout,yout/scale,char(vpa(xout,d)),'fontsize',12,'
        horizontalAlignment','left','Color','g')
514
515
516 [xout,yout] = intersections(t,DRAM_P_iso_F,t,Conv_MRAM_P_iso_F,1);
517 loglog(xout,yout,'.','Color',[1,0.5,0],'markersize',10)
518 text(xout,yout/scale,char(vpa(xout,d)),'fontsize',12,'
        horizontalAlignment','left','Color',[1,0.5,0])
519
520 [xout,yout] = intersections(t,SRAM_P_iso_F,t,Conv_MRAM_P_iso_F,1);
521 loglog(xout,yout,'.','Color',[1,0.5,0],'markersize',10)
522 text(xout,yout*scale,char(vpa(xout,d)),'fontsize',12,'
        horizontalAlignment','right','color',[1,0.5,0])
523
524
525 [xout,yout] = intersections(t,DRAM_P_iso_F,t,STT_MRAM_P_iso_F,1);
526 plot(xout,yout,'r.','markersize',10)
527 text(xout,yout/scale,char(vpa(xout,d)),'fontsize',12,'
        horizontalAlignment','left','color','r')
528
529 [xout,yout] = intersections(t,SRAM_P_iso_F,t,STT_MRAM_P_iso_F,1);
530 plot(xout,yout,'r.','markersize',10)
531 text(xout,yout/scale,char(vpa(xout,d)),'fontsize',12,'
        horizontalAlignment','left','color','r')
532 %
533 %
534 [xout,yout] = intersections(t,DRAM_P_iso_F,t,PCRAM_P_iso_F,1);
535 plot(xout,yout,'m.','markersize',10)
536 text(xout,yout*(scale*1),char(vpa(xout,d)),'fontsize',12,'
        horizontalAlignment','right','color','m')
537
538 [xout,yout] = intersections(t,SRAM_P_iso_F,t,PCRAM_P_iso_F,1);
539 plot(xout,yout,'m.','markersize',10)
540 text(xout,yout*scale,char(vpa(xout,d)),'fontsize',12,'
        horizontalAlignment','right','color','m')
```

```
541
542 if (save)
543 saveas(gcf,'PvsDutycycleiso_F','pdf')
544 end
545
546 if(save)
547    close all
548 end
```

A.2 DRAM

This is the script that calculates the parameters for the DRAM.

```
1 %Area:
2 %Input
3 DRAM_F=[31 18 65]*1E-9;
4 DRAM_Xa=[6 4 4];
5 DRAM_bit_cell=[1 1 1];
6
7 %Output
8 DRAM_A=cell_area(DRAM_F,DRAM_Xa);
9 DRAM_density=bit_density(DRAM_bit_cell,DRAM_A);
10
11 DRAM_w_cell=sqrt(DRAM_A);
12 DRAM_h_cell=DRAM_A./DRAM_w_cell;
13
14
15 DRAM_N_cells=number_of_bytes*8./DRAM_bit_cell;
16 if (N_WL_squareroot)
17     DRAM_N_WL=sqrt(DRAM_N_cells);
18 else
19     DRAM_N_WL=N_WL;
20 end
21 DRAM_N_BL=DRAM_N_cells./DRAM_N_WL;
22
23
24 %Paramaters:
25 %Output
26 DRAM_L_BL=(DRAM_N_BL).*DRAM_w_cell;
27 DRAM_L_WL=(DRAM_N_WL).*DRAM_h_cell;
28
29 %Interconnects:
30 %Input
31 DRAM_eps_r=[3.85 2.65 3.85];
32 DRAM_rho=[2.2 2.2 2.2]*1e-8;
```

```
33 DRAM_AR=[1.8 2 1.7];
34
35 %Output
36 DRAM_C_BL=C_tot(DRAM_L_BL,DRAM_eps_r,DRAM_AR,DRAM_F,DRAM_Xa,
       DRAM_N_BL,new);
37 DRAM_C_WL=C_tot(DRAM_L_WL,DRAM_eps_r,DRAM_AR,DRAM_F,DRAM_Xa,
       DRAM_N_WL,new);
38
39 DRAM_R_BL=R_tot(DRAM_rho,DRAM_L_BL,DRAM_F,DRAM_AR);
40
41 %Write enegry
42 %Input
43 DRAM_C_cell=[25 25 25]*1E-15;
44 DRAM_V_cell=[0.55 0.48 0.65];
45 DRAM_V_BL=DRAM_V_cell;
46
47 %Access transistors
48 %Input
49 DRAM_T_L_g=[27 15.7 38]*1E-9;
50 DRAM_V_WL=[2.7 2.4 3];
51 DRAM_T_C_g=[0.669 0.567 0.791]*1E-9;
52 DRAM_T_R_sd=[467 264 180]*1E-6;
53 %Output
54 DRAM_T_W_g=2*DRAM_T_L_g;
55 DRAM_T_R_FET=DRAM_T_R_sd./DRAM_T_W_g;
56 %Output
57 DRAM_E_write=(0.5).*DRAM_C_cell.*(DRAM_V_cell.^2);
58 DRAM_E_C_BL= (0.5).*DRAM_C_BL.*DRAM_V_BL.^2;
59 DRAM_E_C_WL= (0.5).*DRAM_C_WL.*DRAM_V_WL.^2;
60
61 DRAM_E_T= (0.5).*DRAM_T_C_g.*DRAM_T_W_g.*DRAM_V_WL.^2 ...
62      .*DRAM_N_BL;
63
64
65
66 DRAM_P_write=   Power(DRAM_E_C_WL,DRAM_E_C_BL,DRAM_E_T,
       DRAM_E_write...
67      ,word_size,DRAM_bit_cell,f);
68 DRAM_P_read=   Power(DRAM_E_C_WL,DRAM_E_C_BL,DRAM_E_T,DRAM_E_write
       ...
69      ,word_size,DRAM_bit_cell,f);
70
71 %Refresh
72 DRAM_t_refresh = [64 64 64]*1e-3;
73 DRAM_P_refresh = (DRAM_N_WL./DRAM_t_refresh).*(DRAM_E_C_WL+
       DRAM_E_T+...
74      DRAM_N_BL.*(DRAM_E_C_BL+DRAM_E_write));
75
76 DRAM_P_write_breakdown = [DRAM_E_C_WL.*f;...
```

```
77      DRAM_E_C_BL.*word_size./DRAM_bit_cell.*f;...
78      DRAM_E_write.*word_size.*f;...
79      DRAM_E_T.*f];
80
81  DRAM_P_read_breakdown = [DRAM_E_C_WL.*f;...
82      DRAM_E_C_BL.*word_size./DRAM_bit_cell.*f;...
83      DRAM_E_write.*word_size.*f;...
84      DRAM_E_T.*f];
85
86  %Signal strength
87  DRAM_V_1=(1+DRAM_C_cell./(DRAM_C_cell+DRAM_C_BL)).*DRAM_V_cell;
88  DRAM_V_0=(1-DRAM_C_cell./(DRAM_C_cell+DRAM_C_BL)).*DRAM_V_cell;
89  DRAM_S=(DRAM_V_1-DRAM_V_0)./DRAM_V_1;
90  % Assertions:
91  DRAM_write_test=abs((DRAM_P_write-sum(DRAM_P_write_breakdown))...
92      ./DRAM_P_write);
93  assert(DRAM_write_test(1,1)<1e-9,...
94      'DRAM 2012 write power assertion failed')
95  assert(DRAM_write_test(1,2)<1e-9,...
96      'DRAM 2017 write power assertion failed')
```

A.3 SRAM

This is the script that calculates the parameters for the SRAM.

```
1   %Area:
2   %Input
3   SRAM_F=[32 16.9 65]*1E-9;
4   SRAM_Xa=[140 140 140];
5   SRAM_bit_cell=[1 1 1];
6
7   %Output
8   SRAM_A=cell_area(SRAM_F,SRAM_Xa);
9   SRAM_density=bit_density(SRAM_bit_cell,SRAM_A);
10
11  SRAM_w_cell=sqrt(SRAM_A);
12  SRAM_h_cell=SRAM_A./SRAM_w_cell;
13
14
15  SRAM_N_cells=number_of_bytes*8./SRAM_bit_cell;
16  if (N_WL_squareroot)
17      SRAM_N_WL=sqrt(SRAM_N_cells);
18  else
19      SRAM_N_WL=N_WL;
20  end
```

```
21 SRAM_N_BL=SRAM_N_cells./SRAM_N_WL;
22
23
24 %Paramaters:
25 %Output
26 SRAM_L_BL=(SRAM_N_BL).*SRAM_w_cell;
27 SRAM_L_WL=(SRAM_N_WL).*SRAM_h_cell;
28
29 %Interconnects:
30 %Input
31 SRAM_eps_r=[3.85 2.65 3.85];
32 SRAM_rho=[2.2 2.2 2.2]*1e-8;
33 SRAM_AR=[1.9 2 1.7];
34
35 %Output
36 SRAM_C_BL=2*C_tot(SRAM_L_BL,SRAM_eps_r,SRAM_AR,SRAM_F,SRAM_Xa,
       SRAM_N_BL,new);
37 SRAM_C_WL=C_tot(SRAM_L_WL,SRAM_eps_r,SRAM_AR,SRAM_F,SRAM_Xa,
       SRAM_N_WL,new);
38
39 SRAM_R_BL=R_tot(SRAM_rho,SRAM_L_BL,SRAM_F,SRAM_AR);
40
41 %Write enegry
42 %Input
43 SRAM_CV2=[0.7 0.32 0.96]*1E-9;
44 SRAM_L_g=[27 15.7 38]*1E-9;
45 SRAM_Vdd=[0.9 0.75 1.1];
46
47 SRAM_V_BL=SRAM_Vdd;
48
49
50 %Access transistors
51 %Input
52 SRAM_T_L_g=[27 15.7 38]*1E-9;
53 SRAM_V_WL=[0.9 0.75 1.1];
54 SRAM_T_C_g=[0.866 0.867 0.791]*1E-9;
55 SRAM_T_R_sd=[290 264 180]*1E-6;
56 %Output
57 SRAM_T_W_g=3*SRAM_T_L_g;
58 SRAM_T_R_FET=SRAM_T_R_sd./SRAM_T_W_g;
59
60 %Output
61 SRAM_E_write=(4.5).*SRAM_CV2.*SRAM_L_g;
62 SRAM_E_read=[0 0 0];
63 SRAM_E_C_BL= (0.5).*SRAM_C_BL.*SRAM_V_BL.^2;
64 SRAM_E_C_WL= (0.5).*SRAM_C_WL.*SRAM_V_WL.^2;
65
66 SRAM_E_T= (0.5).*SRAM_T_C_g.*SRAM_T_W_g.*SRAM_V_WL.^2 ...
67     .*SRAM_N_BL.*2;
```

```
68
69 SRAM_P_write=   Power(SRAM_E_C_WL,SRAM_E_C_BL,SRAM_E_T,
    SRAM_E_write...
70     ,word_size,SRAM_bit_cell,f);
71 SRAM_P_read=    Power(SRAM_E_C_WL,SRAM_E_C_BL,SRAM_E_T,SRAM_E_read
    ...
72     ,word_size,SRAM_bit_cell,f);
73 %Retention
74 SRAM_I_leak = [10 10 10]*1e-6;
75 SRAM_P_retention = 4.*SRAM_I_leak.*SRAM_Vdd.*SRAM_L_g.*
    SRAM_N_cells;
76
77 SRAM_P_write_breakdown = [SRAM_E_C_WL.*f;...
78     SRAM_E_C_BL.*word_size./SRAM_bit_cell.*f;...
79     SRAM_E_write.*word_size.*f;...
80     SRAM_E_T.*f];
81 SRAM_P_read_breakdown = [SRAM_E_C_WL.*f;...
82     SRAM_E_C_BL.*word_size./SRAM_bit_cell.*f;...
83     SRAM_E_read.*word_size.*f;...
84     SRAM_E_T.*f];
85 %
86 SRAM_V_th=[637 479 567]*1e-3;
87 SRAM_S=SRAM_V_th./SRAM_Vdd;
88
89 % Assertions:
90 SRAM_write_test=abs((SRAM_P_write-sum(SRAM_P_write_breakdown))...
91     ./SRAM_P_write);
92 assert(SRAM_write_test(1,1)<1e-9,...
93     'SRAM 2012 write power assertion failed')
94 assert(SRAM_write_test(1,2)<1e-9,...
95     'SRAM 2017 write power assertion failed')
```

A.4 FeRAM

This is the script that calculates the parameters for the FeRAM.

```
1 %Area:
2 %Input
3 FeRAM_F=[130 90 65]*1e-9;
4 FeRAM_Xa=[23 22 15];
5 FeRAM_bit_cell=[1 1 1];
6
7 %Output
8 FeRAM_A=cell_area(FeRAM_F,FeRAM_Xa);
9 FeRAM_density=bit_density(FeRAM_bit_cell,FeRAM_A);
```

```
10
11 FeRAM_w_cell=sqrt(FeRAM_A);
12 FeRAM_h_cell=FeRAM_A./FeRAM_w_cell;
13
14 FeRAM_N_cells=number_of_bytes*8./FeRAM_bit_cell;
15 if (N_WL_squareroot)
16     FeRAM_N_WL=sqrt(FeRAM_N_cells);
17 else
18     FeRAM_N_WL=N_WL;
19 end
20 FeRAM_N_BL=FeRAM_N_cells./FeRAM_N_WL;
21
22 %Paramaters:
23 %Output
24 FeRAM_L_BL=(FeRAM_N_BL).*FeRAM_w_cell;
25 FeRAM_L_WL=(FeRAM_N_WL).*FeRAM_h_cell;
26
27 %Interconnects:
28 %Input
29 FeRAM_eps_r=[4.1 3.85 3.85];
30 FeRAM_rho=[3.3 3.3 2.2]*1e-8;
31 FeRAM_AR=[1.6 1.7 1.7]; %2012 values updated with itrs2011
32
33 %Output
34 FeRAM_C_BL=C_tot(FeRAM_L_BL,FeRAM_eps_r,FeRAM_AR,FeRAM_F,FeRAM_Xa,
     FeRAM_N_BL,new);
35 FeRAM_C_WL=C_tot(FeRAM_L_WL,FeRAM_eps_r,FeRAM_AR,FeRAM_F,FeRAM_Xa,
     FeRAM_N_WL,new);
36 FeRAM_C_PL=FeRAM_C_BL;
37
38 FeRAM_R_BL=R_tot(FeRAM_rho,FeRAM_L_BL,FeRAM_F,FeRAM_AR);
39
40 %Write enegry
41 %Input
42 FeRAM_sigma=[8.5 12 0.175]*1e4*1e-6;
43 FeRAM_A_act=[0.423 0.234 0.33]*1E-12;
44 FeRAM_V_cell=[1.5 1.2 1];
45
46 FeRAM_V_BL=FeRAM_V_cell;
47 FeRAM_V_PL=FeRAM_V_cell;
48
49 %Access transistors
50 %Input
51 FeRAM_T_L_g=[120 53 32]*1E-9;
52 FeRAM_V_WL=[1.65 0.9 1.1];
53 FeRAM_T_C_g=[1.669 1.036 0.789]*1E-9;
54 FeRAM_T_R_sd=[200 180 180]*1E-6;
55 %Output
56 FeRAM_T_W_g=2*FeRAM_T_L_g;
```

```
57 FeRAM_T_R_FET=FeRAM_T_R_sd./FeRAM_T_W_g;
58
59 %Output
60 FeRAM_E_write=FeRAM_sigma.*FeRAM_A_act.*FeRAM_V_cell;
61 FeRAM_E_C_BL= (0.5).*FeRAM_C_BL.*FeRAM_V_BL.^2;
62 FeRAM_E_C_WL= (0.5).*FeRAM_C_WL.*FeRAM_V_WL.^2;
63 FeRAM_E_C_PL= (0.5).*FeRAM_C_PL.*FeRAM_V_PL.^2;
64
65 FeRAM_E_T= (0.5).*FeRAM_T_C_g.*FeRAM_T_W_g.*FeRAM_V_WL.^2 ...
66     .*FeRAM_N_BL;
67
68 %FeRAM_P_write=(0.5*FeRAM_E_C_BL+FeRAM_E_C_WL+...
69 %     FeRAM_E_C_PL+FeRAM_E_write).*f;
70 %FeRAM_P_read=     (FeRAM_E_C_WL+FeRAM_E_C_PL+FeRAM_E_write).*f;
71
72 FeRAM_P_write=Power(FeRAM_E_C_WL,0.5*FeRAM_E_C_BL+FeRAM_E_C_PL,
    ...
73     FeRAM_E_T,FeRAM_E_write,word_size,FeRAM_bit_cell,f);
74 % Adding the energy required to charge the bitlines and platelines
     here,
75 % so that it is fits with the function format.
76 FeRAM_P_read=Power(FeRAM_E_C_WL,FeRAM_E_C_BL+FeRAM_E_C_PL, ...
77     FeRAM_E_T,2*FeRAM_E_write,word_size,FeRAM_bit_cell,f);
78
79
80 %Refresh
81 FeRAM_P_refresh = [0 0 0];
82
83 FeRAM_P_write_breakdown = [FeRAM_E_C_WL.*f;...
84     (0.5*FeRAM_E_C_BL+FeRAM_E_C_PL).*word_size./FeRAM_bit_cell.*f
        ;...
85     FeRAM_E_write.*word_size.*f;...
86     FeRAM_E_T.*f];
87 %Again adding the energy required to charge the plateline to the
    bitline
88 %to get the same format as all other memories.
89 FeRAM_P_read_breakdown = [FeRAM_E_C_WL.*f;...
90     (FeRAM_E_C_BL+FeRAM_E_C_PL).*word_size./FeRAM_bit_cell.*f;...
91     FeRAM_E_write.*word_size.*f;...
92     FeRAM_E_T.*f];
93
94 %Signal strength
95 %FeRAM_D_P_sw=[80 80 80]*1e-2;
96 FeRAM_D_P_sw=FeRAM_sigma;
97 eps_PZT=1000*8.854*1e-12;
98 FeRAM_D_P_nsw=eps_PZT*FeRAM_V_cell./(FeRAM_F.*FeRAM_AR);
99
100 FeRAM_V_1=FeRAM_D_P_sw.*FeRAM_A_act./FeRAM_C_BL;
101 FeRAM_V_0=FeRAM_D_P_nsw.*FeRAM_A_act./FeRAM_C_BL;
```

```
102 FeRAM_S=(FeRAM_V_1-FeRAM_V_0)./FeRAM_V_1;
103
104
105 % Assertions:
106 FeRAM_write_test=abs((FeRAM_P_write-sum(FeRAM_P_write_breakdown))
      ...
107     ./FeRAM_P_write);
108 assert(FeRAM_write_test(1,1)<1e-9,...
109     'FeRAM 2012 write power assertion failed')
110 assert(FeRAM_write_test(1,2)<1e-9,...
111     'FeRAM 2017 write power assertion failed')
```

A.5 MRAM

This is the script that calculates the parameters for the conventional MRAM.

```
1 %Area:
2 %Input
3 Conv_MRAM_F=[90 65 65]*1e-9;
4 Conv_MRAM_Xa=[51 52 52];
5 Conv_MRAM_bit_cell=[1 1 1];
6
7 %Output
8 Conv_MRAM_A=cell_area(Conv_MRAM_F,Conv_MRAM_Xa);
9 Conv_MRAM_density=bit_density(Conv_MRAM_bit_cell,Conv_MRAM_A);
10
11 Conv_MRAM_w_cell=sqrt(Conv_MRAM_A);
12 Conv_MRAM_h_cell=Conv_MRAM_A./Conv_MRAM_w_cell;
13
14
15 %Paramaters:
16 %Output
17
18 Conv_MRAM_N_cells=number_of_bytes*8./Conv_MRAM_bit_cell;
19 if (N_WL_squareroot)
20     Conv_MRAM_N_WL=sqrt(Conv_MRAM_N_cells);
21 else
22     Conv_MRAM_N_WL=N_WL;
23 end
24 Conv_MRAM_N_BL=Conv_MRAM_N_cells./Conv_MRAM_N_WL;
25
26 Conv_MRAM_L_BL=(Conv_MRAM_N_BL).*Conv_MRAM_w_cell;
27 Conv_MRAM_L_WL=(Conv_MRAM_N_WL).*Conv_MRAM_h_cell;
28
29 %Interconnects:
```

```
30 %Input
31 Conv_MRAM_eps_r=[3.85 3.85 3.85];
32 Conv_MRAM_rho=[3.3 2.2 2.2]*1e-8;
33 Conv_MRAM_AR=[1.7 1.7 1.7];
34
35 %Output
36 Conv_MRAM_C_BL=C_tot(Conv_MRAM_L_BL,Conv_MRAM_eps_r, Conv_MRAM_AR,
       Conv_MRAM_F,Conv_MRAM_Xa,Conv_MRAM_N_BL,new);
37 Conv_MRAM_C_WL=C_tot(Conv_MRAM_L_WL,Conv_MRAM_eps_r, Conv_MRAM_AR,
       Conv_MRAM_F,Conv_MRAM_Xa,Conv_MRAM_N_WL,new);
38
39
40 Conv_MRAM_R_BL=R_tot(Conv_MRAM_rho,Conv_MRAM_L_BL,Conv_MRAM_F,
       Conv_MRAM_AR);
41
42 %Write enegry
43 %Input
44 Conv_MRAM_E_write=[120 110 110]*1e-12;
45 Conv_MRAM_RA=[1200 600 600]*1e-12;
46 Conv_MRAM_A_act=[0.124 0.066 0.066]*1e-12;
47 Conv_MRAM_TMR=[65 90 90]*1e-2;
48
49 Conv_MRAM_V_read=[250 250 250]*1e-3;
50 Conv_MRAM_t_read=[10 1 1]*1e-9;
51
52 Conv_MRAM_V_BL_read=Conv_MRAM_V_read;
53 Conv_MRAM_V_BL_write=[1.8 1.5 1.5];
54
55
56 %Access transistors
57 %Input
58 Conv_MRAM_T_L_g=[53 32 32]*1E-9;
59 Conv_MRAM_V_WL=[0.9 0.8 0.8];
60 Conv_MRAM_T_C_g=[1.036 0.789 0.789]*1E-9;
61 Conv_MRAM_T_R_sd=[180 190 190]*1E-6;
62 %Output
63 Conv_MRAM_T_W_g=2*Conv_MRAM_T_L_g;
64 Conv_MRAM_T_R_FET=Conv_MRAM_T_R_sd./Conv_MRAM_T_W_g;
65 %Output
66 Conv_MRAM_R_P=Conv_MRAM_RA./Conv_MRAM_A_act;
67 Conv_MRAM_R_AP=Conv_MRAM_R_P.*Conv_MRAM_TMR+Conv_MRAM_R_P;
68
69
70
71 Conv_MRAM_E_C_BL_write=  (0.5).*Conv_MRAM_C_BL.*
       Conv_MRAM_V_BL_write.^2;
72 Conv_MRAM_E_C_BL_read =  (0.5).*Conv_MRAM_C_BL.*
       Conv_MRAM_V_BL_read.^2;
73 Conv_MRAM_E_C_WL=        (0.5).*Conv_MRAM_C_WL.*Conv_MRAM_V_WL.^2;
```

```
74 % Technically this should be called the digit line, but in order
    to stay
75 % true to the naming convention it is called the word line.
76 Conv_MRAM_E_read=        (0.5).*(Conv_MRAM_V_read).^2.*
    Conv_MRAM_t_read.*...
77     ((Conv_MRAM_R_P+Conv_MRAM_R_BL+Conv_MRAM_T_R_FET).^(-1)...
78     +(Conv_MRAM_R_AP+Conv_MRAM_R_BL+Conv_MRAM_T_R_FET).^(-1));
79
80 Conv_MRAM_E_write=Conv_MRAM_E_write./2+Conv_MRAM_E_read;
81
82 Conv_MRAM_E_T= (0.5).*Conv_MRAM_T_C_g.*Conv_MRAM_T_W_g.*
    Conv_MRAM_V_WL.^2 ...
83     .*Conv_MRAM_N_BL;
84
85
86
87 Conv_MRAM_P_read=Power(Conv_MRAM_E_C_WL,Conv_MRAM_E_C_BL_read, ...
88     Conv_MRAM_E_T,Conv_MRAM_E_read,word_size,Conv_MRAM_bit_cell,f)
        ;
89 Conv_MRAM_P_write=Power(Conv_MRAM_E_C_BL_write,
    Conv_MRAM_E_C_BL_write, ...
90     [0 , 0 , 0],Conv_MRAM_E_write,word_size,Conv_MRAM_bit_cell,f)
        ...
91     +Conv_MRAM_P_read;
92 %Adding read energy due to toggle, and set 0 0 as the access
    transistor
93 %energy.
94
95 %Refresh
96 Conv_MRAM_P_refresh = [0 0 0];
97
98
99 Conv_MRAM_P_write_breakdown = [Conv_MRAM_E_C_BL_write.*f;...
100     Conv_MRAM_E_C_BL_write.*word_size./Conv_MRAM_bit_cell.*f;...
101     Conv_MRAM_E_write.*word_size.*f;...
102     [0,0,0].*f];
103 Conv_MRAM_P_read_breakdown = [Conv_MRAM_E_C_WL.*f;...
104     Conv_MRAM_E_C_BL_read.*word_size./Conv_MRAM_bit_cell.*f;...
105     Conv_MRAM_E_read.*word_size.*f;...
106     Conv_MRAM_E_T.*f];
107
108 Conv_MRAM_I_1=(Conv_MRAM_V_read)./...
109     (Conv_MRAM_R_P+Conv_MRAM_R_BL+Conv_MRAM_T_R_FET);
110 Conv_MRAM_I_0=(Conv_MRAM_V_read)./...
111     (Conv_MRAM_R_AP+Conv_MRAM_R_BL+Conv_MRAM_T_R_FET);
112 Conv_MRAM_S=(Conv_MRAM_I_1-Conv_MRAM_I_0)./Conv_MRAM_I_1;
113 % Assertions:
114 Conv_MRAM_write_test=abs((Conv_MRAM_P_write-sum(
    Conv_MRAM_P_write_breakdown)-Conv_MRAM_P_read)...
```

```
115     ./Conv_MRAM_P_write);
116 assert(Conv_MRAM_write_test(1,1)<1e-9,...
117     'Conv_MRAM 2012 write power assertion failed')
118 assert(Conv_MRAM_write_test(1,2)<1e-9,...
119     'Conv_MRAM 2017 write power assertion failed')
```

A.6 STT-MRAM

This is the script that calculates the parameters for the STT-MRAM.

```
1 %Area:
2 %Input
3 STT_MRAM_F=[65 32 65]*1e-9;
4 STT_MRAM_Xa=[20 10 20];
5 STT_MRAM_bit_cell=[1 1 1];
6
7 %Output
8 STT_MRAM_A=cell_area(STT_MRAM_F,STT_MRAM_Xa);
9 STT_MRAM_density=bit_density(STT_MRAM_bit_cell,STT_MRAM_A);
10
11 STT_MRAM_w_cell=sqrt(STT_MRAM_A);
12 STT_MRAM_h_cell=STT_MRAM_A./STT_MRAM_w_cell;
13
14
15 %Paramaters:
16 %Output
17
18 STT_MRAM_N_cells=number_of_bytes*8./STT_MRAM_bit_cell;
19 if (N_WL_squareroot)
20     STT_MRAM_N_WL=sqrt(STT_MRAM_N_cells);
21 else
22     STT_MRAM_N_WL=N_WL;
23 end
24 STT_MRAM_N_BL=STT_MRAM_N_cells./STT_MRAM_N_WL;
25
26 STT_MRAM_L_BL=(STT_MRAM_N_BL).*STT_MRAM_w_cell;
27 STT_MRAM_L_WL=(STT_MRAM_N_WL).*STT_MRAM_h_cell;
28
29 %Interconnects:
30 %Input
31 STT_MRAM_eps_r=[3.85 3.85 3.85];
32 STT_MRAM_rho=[2.2 2.2 2.2]*1e-8;
33 STT_MRAM_AR=[1.7 1.8 1.7];
34
35 %Output
```

```
36 STT_MRAM_C_BL=C_tot(STT_MRAM_L_BL,STT_MRAM_eps_r, STT_MRAM_AR,
       STT_MRAM_F,STT_MRAM_Xa,STT_MRAM_N_BL,new);
37 STT_MRAM_C_WL=C_tot(STT_MRAM_L_WL,STT_MRAM_eps_r, STT_MRAM_AR,
       STT_MRAM_F,STT_MRAM_Xa,STT_MRAM_N_WL,new);
38
39
40 STT_MRAM_R_BL=R_tot(STT_MRAM_rho,STT_MRAM_L_BL,STT_MRAM_F,
       STT_MRAM_AR);
41
42 %Write enegry
43 %Input
44 STT_MRAM_E_write=[2.2 0.3 2.2]*1e-12;
45 STT_MRAM_RA=[11 10 11]*1e-12;
46 STT_MRAM_A_act=[0.008 0.003 0.008]*1e-12;
47 STT_MRAM_TMR=[120 150 120]*1e-2;
48
49 STT_MRAM_V_read=[250 250 250]*1e-3;
50 STT_MRAM_t_read=[10 1 10]*1e-9;
51
52 STT_MRAM_V_BL_read=STT_MRAM_V_read;
53 STT_MRAM_V_BL_write=[1.8 1.5 1.8];
54
55
56 %Output
57 STT_MRAM_R_P=STT_MRAM_RA./STT_MRAM_A_act;
58 STT_MRAM_R_AP=STT_MRAM_R_P.*STT_MRAM_TMR+STT_MRAM_R_P;
59
60 %Access transistors
61 %Input
62 STT_MRAM_T_L_g=[29 22 29]*1E-9;
63 STT_MRAM_V_WL=[1.1 0.87 1.1];
64 STT_MRAM_T_C_g=[0.721 0.752 0.721]*1E-9;
65 STT_MRAM_T_R_sd=[200 366 200]*1E-6;
66 %Output
67 STT_MRAM_T_W_g=[174 44 174]*1E-9;
68 STT_MRAM_T_R_FET=STT_MRAM_T_R_sd./STT_MRAM_T_W_g;
69
70 STT_MRAM_E_C_BL_write=  (0.5).*STT_MRAM_C_BL.*STT_MRAM_V_BL_write
       .^2;
71 STT_MRAM_E_C_BL_read =  (0.5).*STT_MRAM_C_BL.*STT_MRAM_V_BL_read
       .^2;
72 STT_MRAM_E_C_WL=        (0.5).*STT_MRAM_C_WL.*STT_MRAM_V_WL.^2;
73
74 STT_MRAM_E_read=        (0.5).*(STT_MRAM_V_read).^2.*
       STT_MRAM_t_read.*...
75      ((STT_MRAM_R_P+STT_MRAM_R_BL+STT_MRAM_T_R_FET).^(-1)...
76      +(STT_MRAM_R_AP+STT_MRAM_R_BL+STT_MRAM_T_R_FET).^(-1));
77
78 STT_MRAM_E_T= (0.5).*STT_MRAM_T_C_g.*STT_MRAM_T_W_g.*STT_MRAM_V_WL
```

```
     .^2 ...
79     .*STT_MRAM_N_BL;
80
81
82
83 STT_MRAM_P_write=Power(STT_MRAM_E_C_WL,STT_MRAM_E_C_BL_write, ...
84     STT_MRAM_E_T,STT_MRAM_E_write,word_size,STT_MRAM_bit_cell,f);
85 STT_MRAM_P_read=Power(STT_MRAM_E_C_WL,STT_MRAM_E_C_BL_read, ...
86     STT_MRAM_E_T,STT_MRAM_E_read,word_size,STT_MRAM_bit_cell,f);
87
88
89 %Refresh
90 STT_MRAM_P_refresh = [0 0 0];
91
92 STT_MRAM_P_write_breakdown = [STT_MRAM_E_C_WL.*f;...
93     STT_MRAM_E_C_BL_write.*word_size./STT_MRAM_bit_cell.*f;...
94     STT_MRAM_E_write.*word_size.*f;...
95     STT_MRAM_E_T.*f];
96 STT_MRAM_P_read_breakdown = [STT_MRAM_E_C_WL.*f;...
97     STT_MRAM_E_C_BL_read.*word_size./STT_MRAM_bit_cell.*f;...
98     STT_MRAM_E_read.*word_size.*f;...
99     STT_MRAM_E_T.*f];
100
101 STT_MRAM_I_1=(STT_MRAM_V_read)./...
102     (STT_MRAM_R_P+STT_MRAM_R_BL+STT_MRAM_T_R_FET);
103 STT_MRAM_I_0=(STT_MRAM_V_read)./...
104     (STT_MRAM_R_AP+STT_MRAM_R_BL+STT_MRAM_T_R_FET);
105 STT_MRAM_S=(STT_MRAM_I_1-STT_MRAM_I_0)./STT_MRAM_I_1;
106
107 % Assertions:
108 STT_MRAM_write_test=abs((STT_MRAM_P_write-sum(
     STT_MRAM_P_write_breakdown))...
109     ./STT_MRAM_P_write);
110 assert(STT_MRAM_write_test(1,1)<1e-9,...
111     'STT_MRAM 2012 write power assertion failed')
112 assert(STT_MRAM_write_test(1,2)<1e-9,...
113     'STT_MRAM 2017 write power assertion failed')
```

A.7 PCRAM

This is the script that calculates the parameters for the PCRAM.

```
1 %Area:
2 %Input
3 PCRAM_F=[38 18 65]*1e-9;
```

```
4 PCRAM_Xa=[12 6 16];
5 PCRAM_bit_cell=[1 4 1];
6
7 %Output
8 PCRAM_A=cell_area(PCRAM_F,PCRAM_Xa);
9 PCRAM_density=bit_density(PCRAM_bit_cell,PCRAM_A);
10
11 PCRAM_w_cell=sqrt(PCRAM_A);
12 PCRAM_h_cell=PCRAM_A./PCRAM_w_cell;
13
14
15 %Paramaters:
16 %Output
17
18 PCRAM_N_cells=number_of_bytes*8./PCRAM_bit_cell;
19 if (N_WL_squareroot)
20     PCRAM_N_WL=sqrt(PCRAM_N_cells);
21 else
22     PCRAM_N_WL=N_WL;
23 end
24 PCRAM_N_BL=PCRAM_N_cells./PCRAM_N_WL;
25
26 PCRAM_L_BL=(PCRAM_N_BL).*PCRAM_w_cell;
27 PCRAM_L_WL=(PCRAM_N_WL).*PCRAM_h_cell;
28
29 %Interconnects:
30 %Input
31 PCRAM_eps_r=[3.85 2.65 3.85];
32 PCRAM_rho=[2.2 2.2 2.2]*1e-8;
33 PCRAM_AR=[1.7 2 1.7];
34
35 %Output
36 PCRAM_C_BL=C_tot(PCRAM_L_BL,PCRAM_eps_r, PCRAM_AR,PCRAM_F,PCRAM_Xa
    ,PCRAM_N_BL,new);
37 PCRAM_C_WL=C_tot(PCRAM_L_WL,PCRAM_eps_r, PCRAM_AR,PCRAM_F,PCRAM_Xa
    ,PCRAM_N_WL,new);
38
39 Fe_RAM_C_PL=PCRAM_C_BL;
40
41 PCRAM_R_BL=R_tot(PCRAM_rho,PCRAM_L_BL,PCRAM_F,PCRAM_AR);
42
43 %Write enegry
44 %Input
45 PCRAM_Ir=[174 57 202]*1e-6;
46
47 PCRAM_tr=[10 6 10]*1e-9;
48 PCRAM_ts=[90 75 100]*1e-9;
49 PCRAM_Rr=[300 1000 300]*1e3;
50 PCRAM_Rs=[2.6 5.5 6]*1e3;
```

```
51 PCRAM_V_read=[200 200 200]*1e-3;
52
53 PCRAM_t_read=[12 2 12]*1e-9;
54
55 PCRAM_V_BL_read=PCRAM_V_read;
56 PCRAM_Is=PCRAM_Ir/sqrt(3);
57
58
59 %Access transistors
60 %Input
61 PCRAM_T_L_g=[24 14 29]*1E-9;
62 PCRAM_V_WL=[0.9 0.75 1.1];
63 PCRAM_T_C_g=[0.936 0.611 0.721]*1E-9;
64 PCRAM_T_R_sd=[330 235 200]*1E-6;
65 %Output
66 PCRAM_T_W_g=[170 33 208]*1E-9;
67 PCRAM_T_R_FET=PCRAM_T_R_sd./PCRAM_T_W_g;
68
69 %Output
70
71 PCRAM_Er=(PCRAM_Ir.^2).*(PCRAM_Rs+PCRAM_R_BL+PCRAM_T_R_FET).*
       PCRAM_tr;
72 PCRAM_Es=(PCRAM_Is.^2).*(PCRAM_Rs+PCRAM_R_BL+PCRAM_T_R_FET).*
       PCRAM_ts;
73 PCRAM_V_BL_write_r=PCRAM_Ir.*(PCRAM_R_BL+PCRAM_T_R_FET+PCRAM_Rs);
74 PCRAM_V_BL_write_s=PCRAM_Ir.*(PCRAM_R_BL+PCRAM_T_R_FET+PCRAM_Rs);
75
76 % Initializing.
77 PCRAM_E_write=zeros(1,3);
78 PCRAM_E_C_BL_write=zeros(1,3);
79 PCRAM_E_C_BL_read=zeros(1,3);
80 PCRAM_E_C_WL_write=zeros(1,3);
81 PCRAM_E_C_WL_read=zeros(1,3);
82 PCRAM_E_read=zeros(1,3);
83 PCRAM_E_T_write=zeros(1,3);
84 PCRAM_E_T_read=zeros(1,3);
85
86 PCRAM_I_1=ones(1,3);
87 PCRAM_I_0=ones(1,3);
88 PCRAM_S=ones(1,3);
89 for j=1:3
90     if(PCRAM_bit_cell(j)==1) % If single level cell
91
92         PCRAM_E_write(j)=(0.5).*(PCRAM_Er(j)+PCRAM_Es(j));
93         PCRAM_E_C_BL_write(j)=(0.5).*PCRAM_C_BL(j).*...
94              (PCRAM_V_BL_write_r(j).^2 +PCRAM_V_BL_write_s(j).^2);
95         PCRAM_E_C_BL_read(j)  =  (0.5).*PCRAM_C_BL(j).*  ...
96             PCRAM_V_BL_read(j).^2;
97         PCRAM_E_C_WL_write(j)=  (0.5).*PCRAM_C_WL(j).*PCRAM_V_WL(j)
```

```
        .^2;
98      PCRAM_E_C_WL_read(j)=  (0.5).*PCRAM_C_WL(j).*PCRAM_V_WL(j)
        .^2;
99      PCRAM_E_read(j)=  (0.5).*(PCRAM_V_read(j)).^2.*PCRAM_t_read
        (j).*...
100         ((PCRAM_Rs(j)+PCRAM_R_BL(j)+PCRAM_T_R_FET(j)).^(-1)...
101         +(PCRAM_Rr(j)+PCRAM_R_BL(j)+PCRAM_T_R_FET(j)).^(-1));
102     PCRAM_E_T_write(j)= (0.5).*PCRAM_T_C_g(j).*PCRAM_T_W_g(j)
        ...
103         .*PCRAM_V_WL(j).^2.*PCRAM_N_BL(j);
104     PCRAM_E_T_read(j)= (0.5).*PCRAM_T_C_g(j).*PCRAM_T_W_g(j)...
105         .*PCRAM_V_WL(j).^2.*PCRAM_N_BL(j);
106
107     PCRAM_I_1(j)=(PCRAM_V_read(j))./...
108         (PCRAM_Rs(j)+PCRAM_R_BL(j)+PCRAM_T_R_FET(j));
109     PCRAM_I_0(j)=(PCRAM_V_read(j))./...
110         (PCRAM_Rr(j)+PCRAM_R_BL(j)+PCRAM_T_R_FET(j));
111     PCRAM_S(j)=(PCRAM_I_1(j)-PCRAM_I_0(j))./PCRAM_I_1(j);
112   else % If multi level cell
113     %defining n as number of levels
114     n=PCRAM_bit_cell(j).^2;
115     i=1:n;
116
117     PCRAM_R_MLC=10.^(log10(PCRAM_Rs(j))+(i-1)*((log10(PCRAM_Rr(
        j))...
118         -log10(PCRAM_Rs(j)))/(n-1)));
119     N_average=15;
120
121     %Code for E_write per bit
122     Delta_I=(PCRAM_Ir(j)-PCRAM_Is(j))/(2*N_average);
123     PCRAM_I_i=PCRAM_Is(j)+((1:N_average)-1)*Delta_I;
124     PCRAM_E_write_i=(PCRAM_Rs(j)+PCRAM_R_BL(j)+PCRAM_T_R_FET(j)
        )...
125         .*PCRAM_tr(j).*PCRAM_I_i.^2;
126     PCRAM_E_read_i=  (PCRAM_V_read(j)).^2.*PCRAM_t_read(j).*...
127         (PCRAM_R_MLC+PCRAM_R_BL(j)+PCRAM_T_R_FET(j)).^(-1);
128     PCRAM_E_read_MLC=log2(n)/n*sum(PCRAM_E_read_i);
129     PCRAM_E_read(j)=PCRAM_E_read_MLC/log2(n);
130
131     PCRAM_E_write(j)=1/log2(n)*(PCRAM_E_read_MLC+(n-1)/n*(
        PCRAM_Es(j)...
132         +(n-1)/n*(PCRAM_Er(j)+(n-2)/n*(N_average*
            PCRAM_E_read_MLC ...
133         +sum(PCRAM_E_write_i)))));
134
135     PCRAM_I_read_i=  PCRAM_V_read(j)./...
136         (PCRAM_R_MLC+PCRAM_R_BL(j)+PCRAM_T_R_FET(j));
137     PCRAM_S(j)=min(-diff(PCRAM_I_read_i)./PCRAM_I_read_i(1:n-1)
        );
```

```
138         %Code for C_BL energy
139         PCRAM_V_BL_i=PCRAM_I_i*(PCRAM_Rs(j)+PCRAM_R_BL(j)+
                PCRAM_T_R_FET(j));
140         PCRAM_E_C_BL_write_i=(0.5).*PCRAM_C_BL(j).*PCRAM_V_BL_i.^2;
141         PCRAM_E_C_BL_s=(0.5).*PCRAM_C_BL(j).*PCRAM_V_BL_write_s(j)
                .^2;
142         PCRAM_E_C_BL_r=(0.5).*PCRAM_C_BL(j).*PCRAM_V_BL_write_r(j)
                .^2;
143
144         % Need log2(n) reads but read log2(n) bits
145         PCRAM_E_C_BL_read(j)=(0.5).*PCRAM_C_BL(j).* ...
146             PCRAM_V_BL_read(j).^2;
147
148
149         PCRAM_E_C_BL_write(j)=1/log2(n)*(log2(n)*PCRAM_E_C_BL_read(
                j)+...
150             (n-1)/n*(PCRAM_E_C_BL_s...
151             +(n-1)/n*(PCRAM_E_C_BL_r...
152             +(n-2)/n*(N_average*PCRAM_E_C_BL_read(j)*log2(n) ...
153             +sum(PCRAM_E_C_BL_write_i)))));
154
155         %Code for C_WL energy and Transitor
156         PCRAM_E_C_WL_read(j)=  (0.5).*PCRAM_C_WL(j).*PCRAM_V_WL(j)
                .^2;
157         PCRAM_E_T_read(j)= (0.5).*PCRAM_T_C_g(j).*PCRAM_T_W_g(j)...
158             .*PCRAM_V_WL(j).^2.*PCRAM_N_BL(j);
159         % This word line read energy is the same energy which is
                needed for
160         % all operations during the write algorihm. Similar for the
161         % transistor energy
162         PCRAM_E_C_WL_write(j)=1/log2(n)*(log2(n)*PCRAM_E_C_WL_read(
                j)+...
163             (n-1)/n*(PCRAM_E_C_WL_read(j)...
164             +(n-1)/n*(PCRAM_E_C_WL_read(j)...
165             +(n-2)/n*(N_average*PCRAM_E_C_WL_read(j)*log2(n) ...
166             +PCRAM_E_C_WL_read(j)*N_average))));
167         PCRAM_E_T_write(j)=1/log2(n)*(log2(n)*PCRAM_E_T_read(j)+...
168             (n-1)/n*(PCRAM_E_T_read(j)...
169             +(n-1)/n*(PCRAM_E_T_read(j)...
170             +(n-2)/n*(N_average*PCRAM_E_T_read(j)*log2(n) ...
171             +PCRAM_E_T_read(j)*N_average))));
172     end
173 end
174
175
176 PCRAM_P_write=Power(PCRAM_E_C_WL_write,PCRAM_E_C_BL_write, ...
177     PCRAM_E_T_write,PCRAM_E_write,word_size,PCRAM_bit_cell,f);
178 PCRAM_P_read=Power(PCRAM_E_C_WL_read,PCRAM_E_C_BL_read, ...
179     PCRAM_E_T_read,PCRAM_E_read,word_size,PCRAM_bit_cell,f);
```

```
180
181
182 %Refresh
183 PCRAM_P_refresh = [0 0 0];
184
185 PCRAM_P_write_breakdown = [PCRAM_E_C_WL_write.*f;...
186     PCRAM_E_C_BL_write.*word_size./PCRAM_bit_cell.*f;...
187     PCRAM_E_write.*word_size.*f;...
188     PCRAM_E_T_write.*f];
189 PCRAM_P_read_breakdown = [PCRAM_E_C_WL_read.*f;...
190     PCRAM_E_C_BL_read.*word_size./PCRAM_bit_cell.*f;...
191     PCRAM_E_read.*word_size.*f;...
192     PCRAM_E_T_read.*f];
193
194 % Assertions:
195 PCRAM_write_test=abs((PCRAM_P_write-sum(PCRAM_P_write_breakdown))
    ...
196     ./PCRAM_P_write);
197 assert(PCRAM_write_test(1,1)<1e-9,...
198     'PCRAM 2012 write power assertion failed')
199 assert(PCRAM_write_test(1,2)<1e-9,...
200     'PCRAM 2017 write power assertion failed')
```

A.8 Functions

This is an inclusions of the functions developed, which are called by the scripts.

```
1 function density=bit_density(bit_cell,A)
2 density=bit_cell./A;
3 end
```

```
1 function C=C_tot(L,eps_r,AR,F,X_AF,N,new)
2 if(new)
3
4     C=2.24.*(N./128).*(F/1e-9).^(0.6).*sqrt(X_AF/4)*1e-15;
5 else
6     eps_0 = 8.854187817620*1E-12;
7     C=2*eps_0.*eps_r.*L.*(AR+AR.^(-1));
8 end
9 end
```

```
1 function R=R_tot(rho,L,F,AR)
```

```
R=rho.*L./(F.^2.*AR);
end
```

```
function P=Power(E_C_WL,E_C_BL,E_T,E,word,bit_cell,frequency)
    P=(E_C_WL+E_T+word.*(E_C_BL./bit_cell+E)).*frequency;
end
```

```
function []= barnumber(plot,numbers_of_plot)
d=2; % number of digits
f=1.25; %scaling over bar
min_height=1e-2;
single=min(size(plot));
if(single==1)
   for i=1:numbers_of_plot
        offset=0;
        if(plot(i,1)<min_height)
            offset=min_height;
        end
        text(i,plot(i,1)*f+offset*12,char(vpa(plot(i,1),d))...
            ,'horizontalAlignment','center','color',[0 0 0.5])
    end
else
    for i=1:numbers_of_plot
        offset=0;
        if(plot(i,1)<min_height)
            offset=min_height;
        end
        text(i-0.05,plot(i,1)*f+offset*12,char(vpa(plot(i,1),d))
           ...
            ,'horizontalAlignment','right','color',[0 0 0.5])
        offset=0;
        if(plot(i,2)<min_height)
            offset=min_height;
        end
        text(i+0.05,plot(i,2)*f+offset*12,char(vpa(plot(i,2),d))
           ...
            ,'horizontalAlignment','left','color',[0.5 0 0])
    end
end
end
```

B Appendix: Schematics

This appendix shows the schematics which where developed during this work. The schematics show how the signals on the PCBs, which were to be designed by Instrumentation lab at NTNU, should be routed.

B.1 MCU Schematics

Schematic for the microcontroller in case study is shown in figure B.1, with corresponding pin-out shown in figure B.2.

B.2 RAM Schematics

Schematics for how each of the RAM chips in the case study should be connected to the RAM socket are shown in figures B.3, B.3, B.4, B.5, B.6, 4.3, B.8 and B.9.

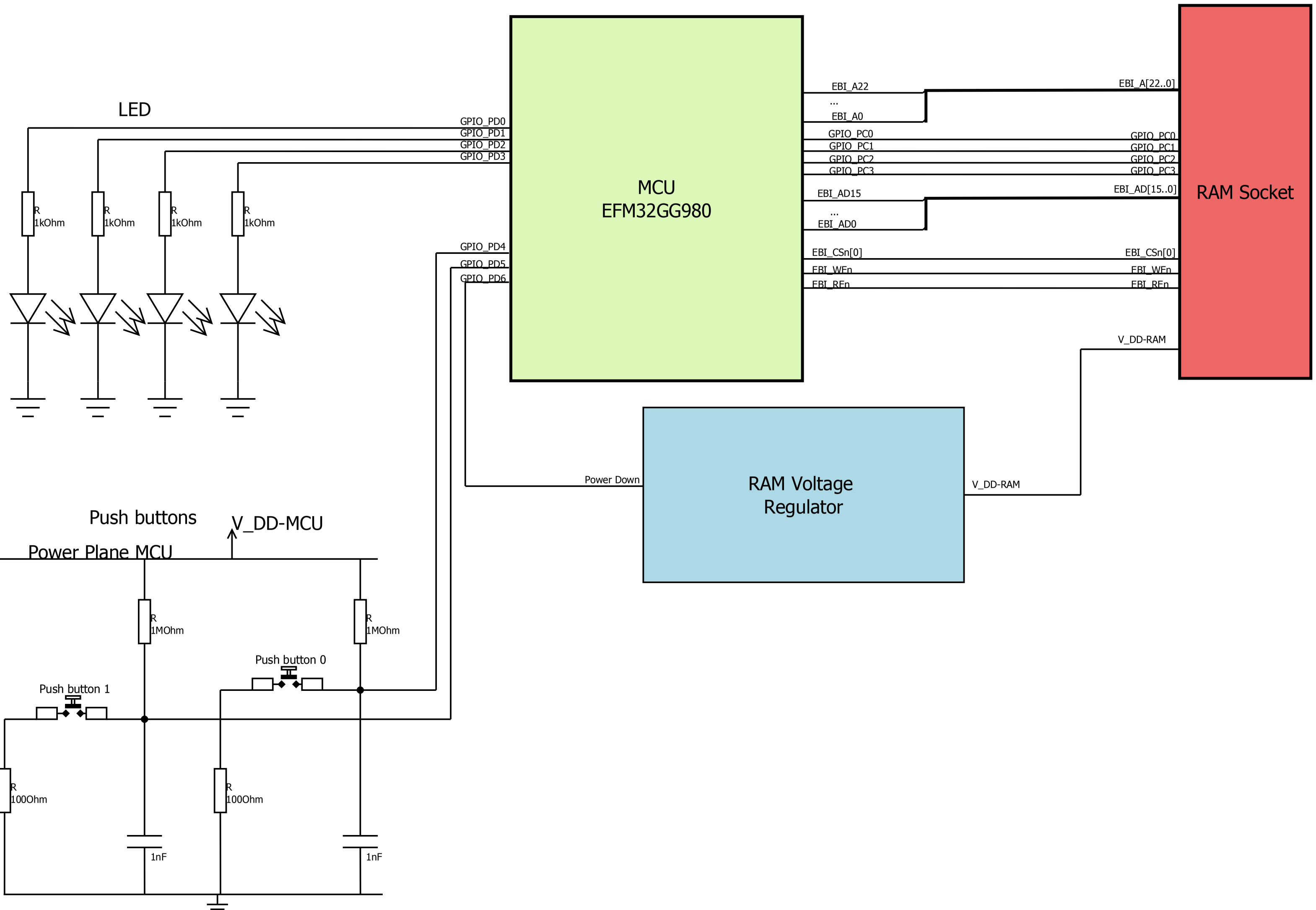


Figure B.1: Schematic of the microcontroller (MCU) printed circuit board (PCB), the figure shows how the different signals from the MCU should be routed. Power supply and oscillators for the microcontroller and corresponding decoupling not shown.

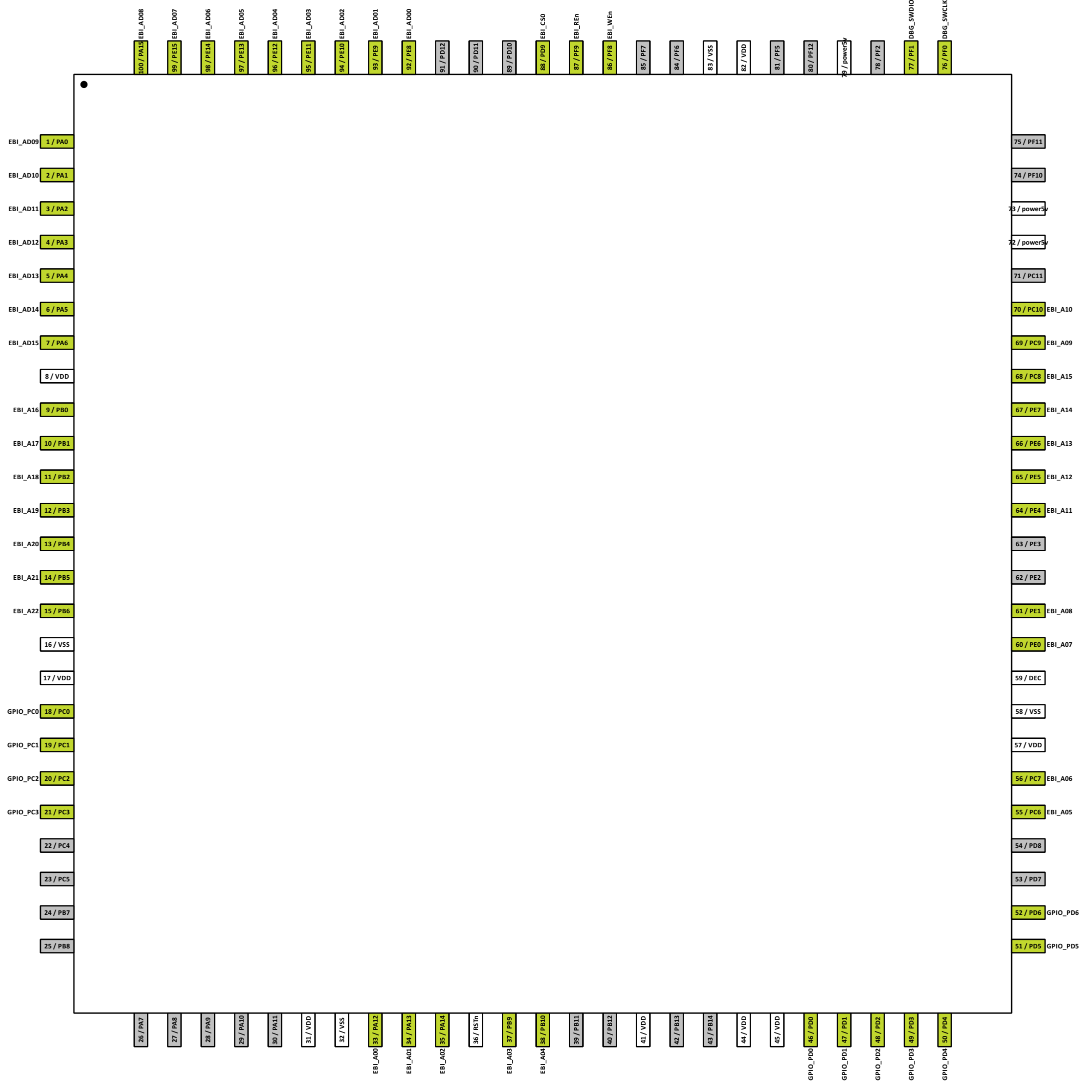


Figure B.2: Microcontroller pin-out, used pins are marked green. Created with the energyAware Designer from Energy Micro AS.

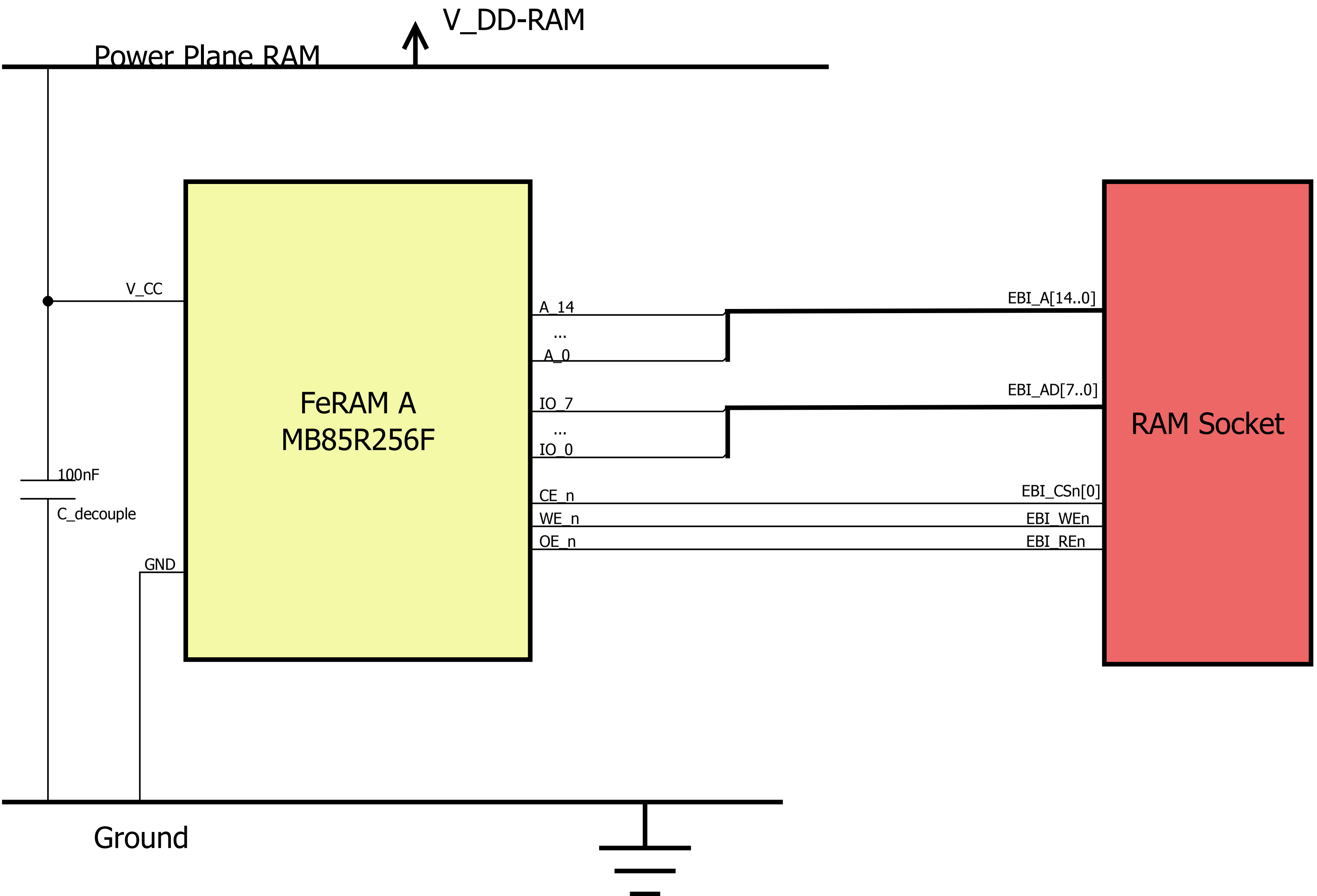


Figure B.3: Schematic of FeRAM A.

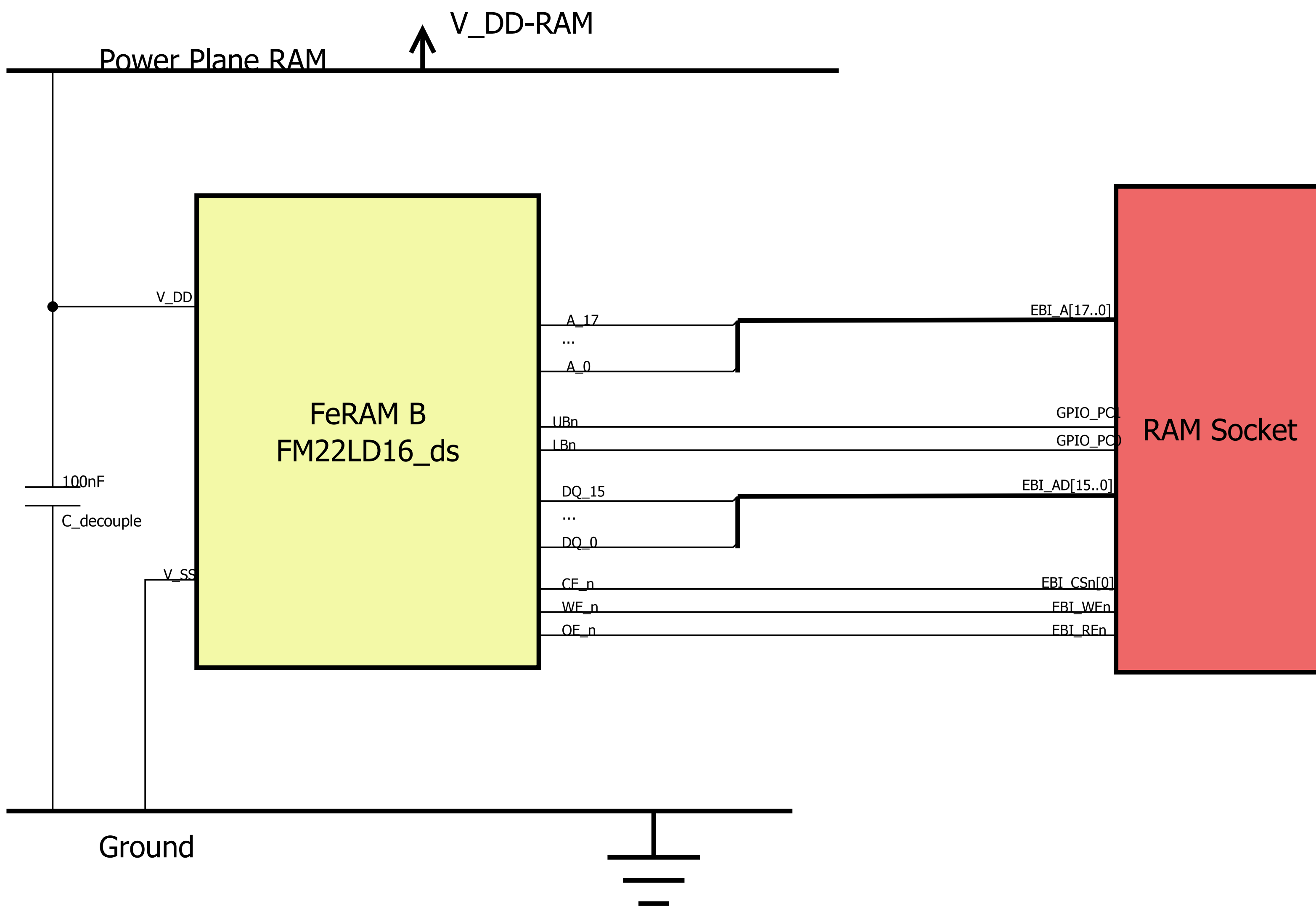


Figure B.4: Schematic of FeRAM B.

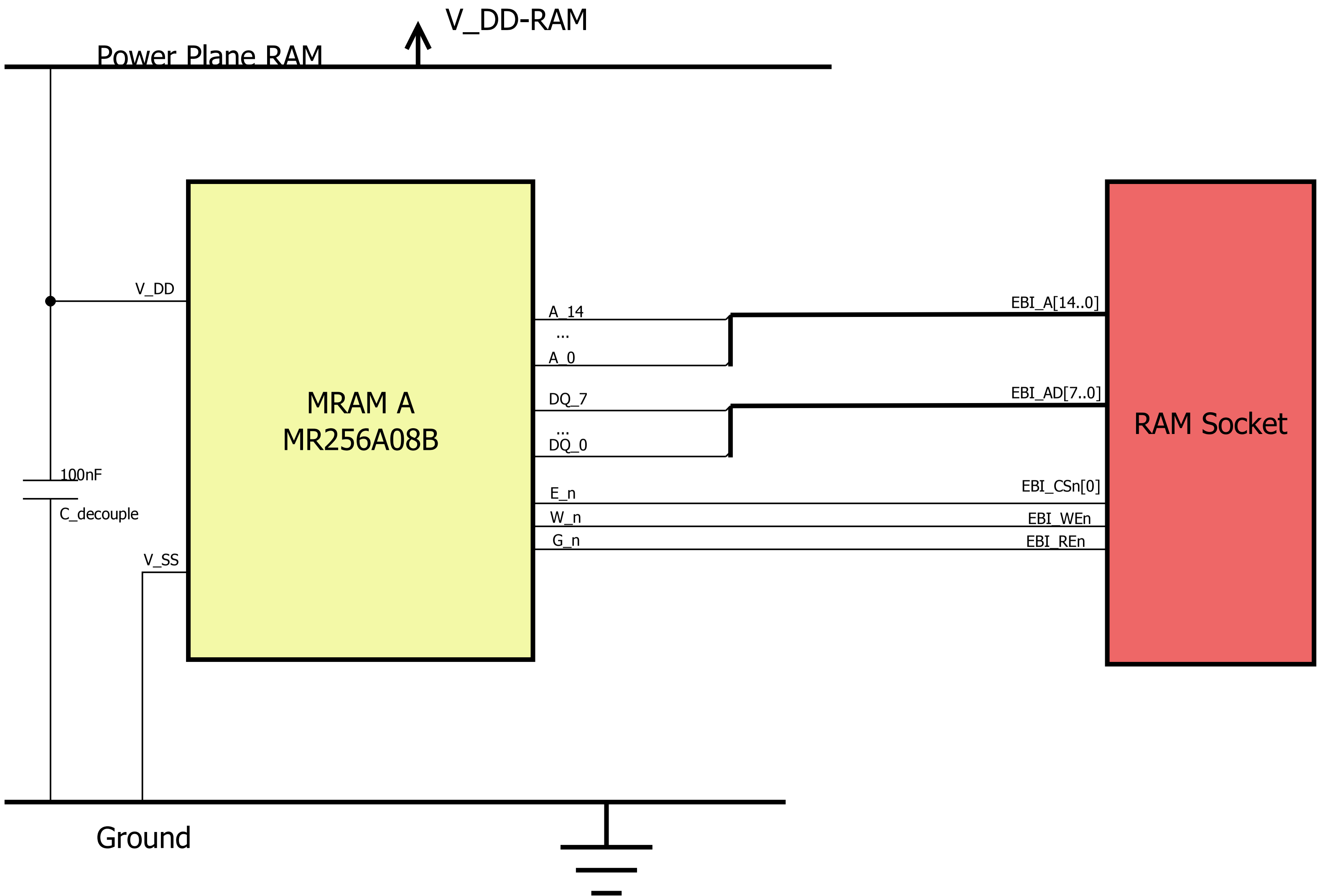


Figure B.5: Schematic of MRAM A.

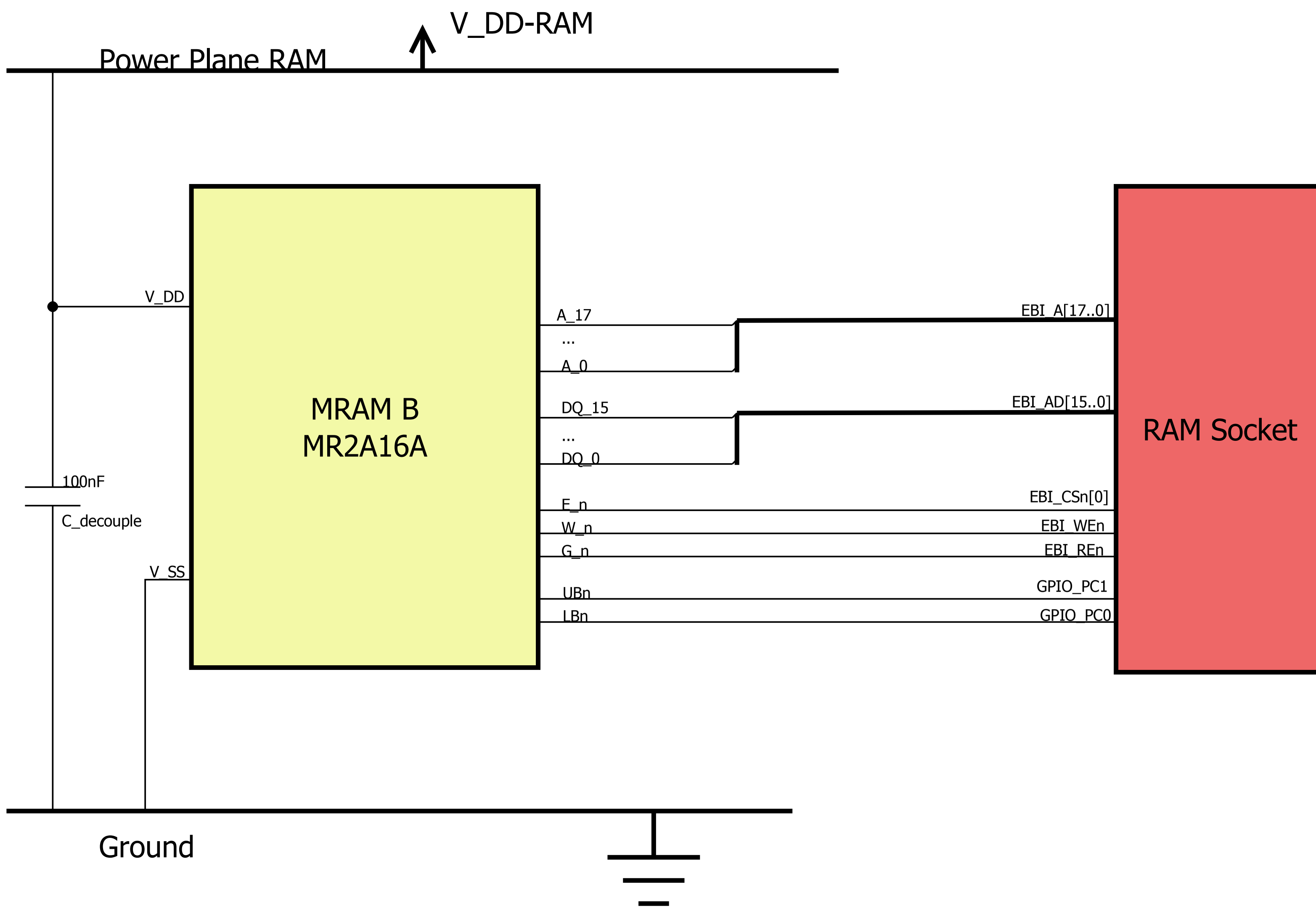


Figure B.6: Schematic of MRAM B.

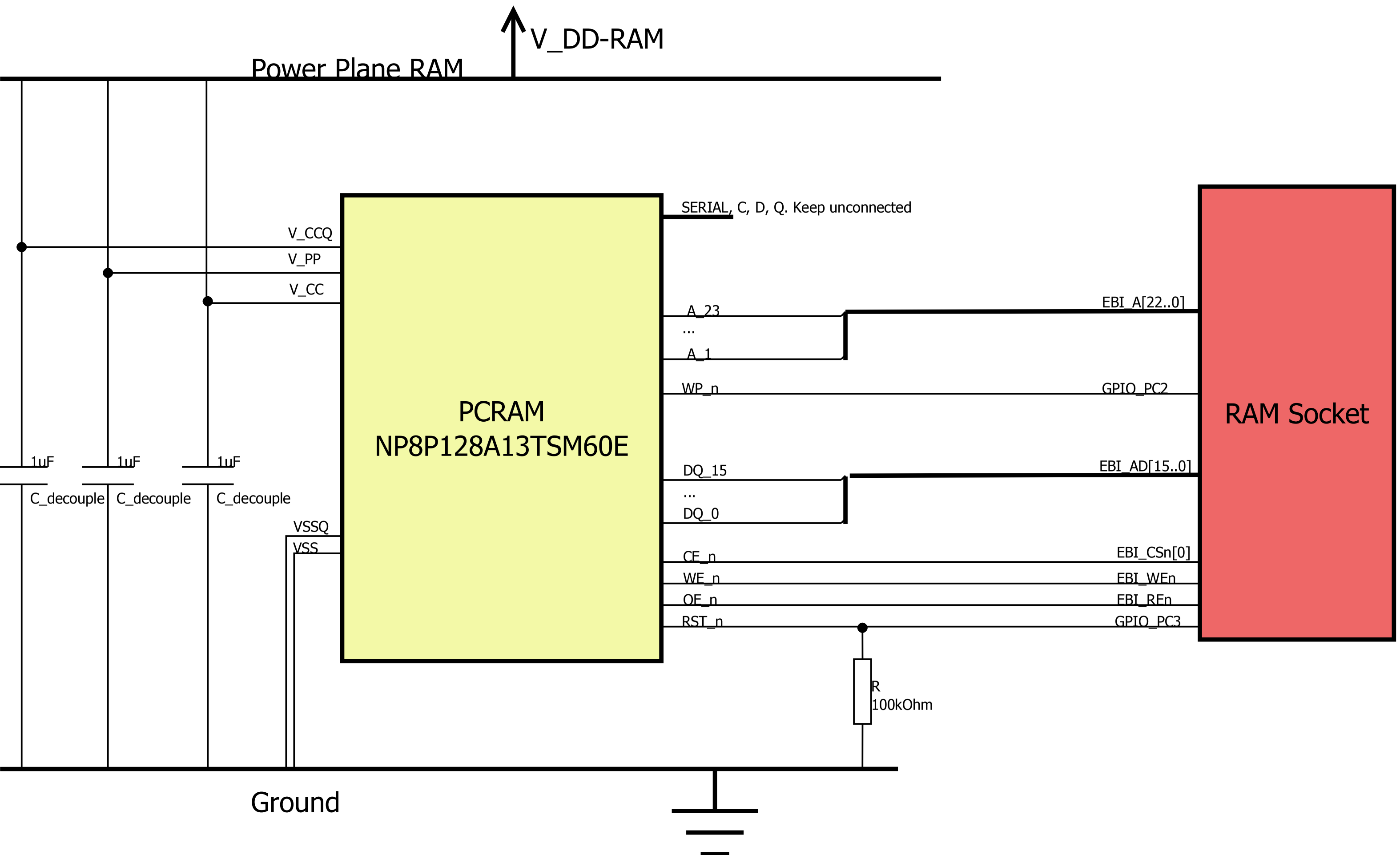


Figure B.7: Schematic of PCRAM.

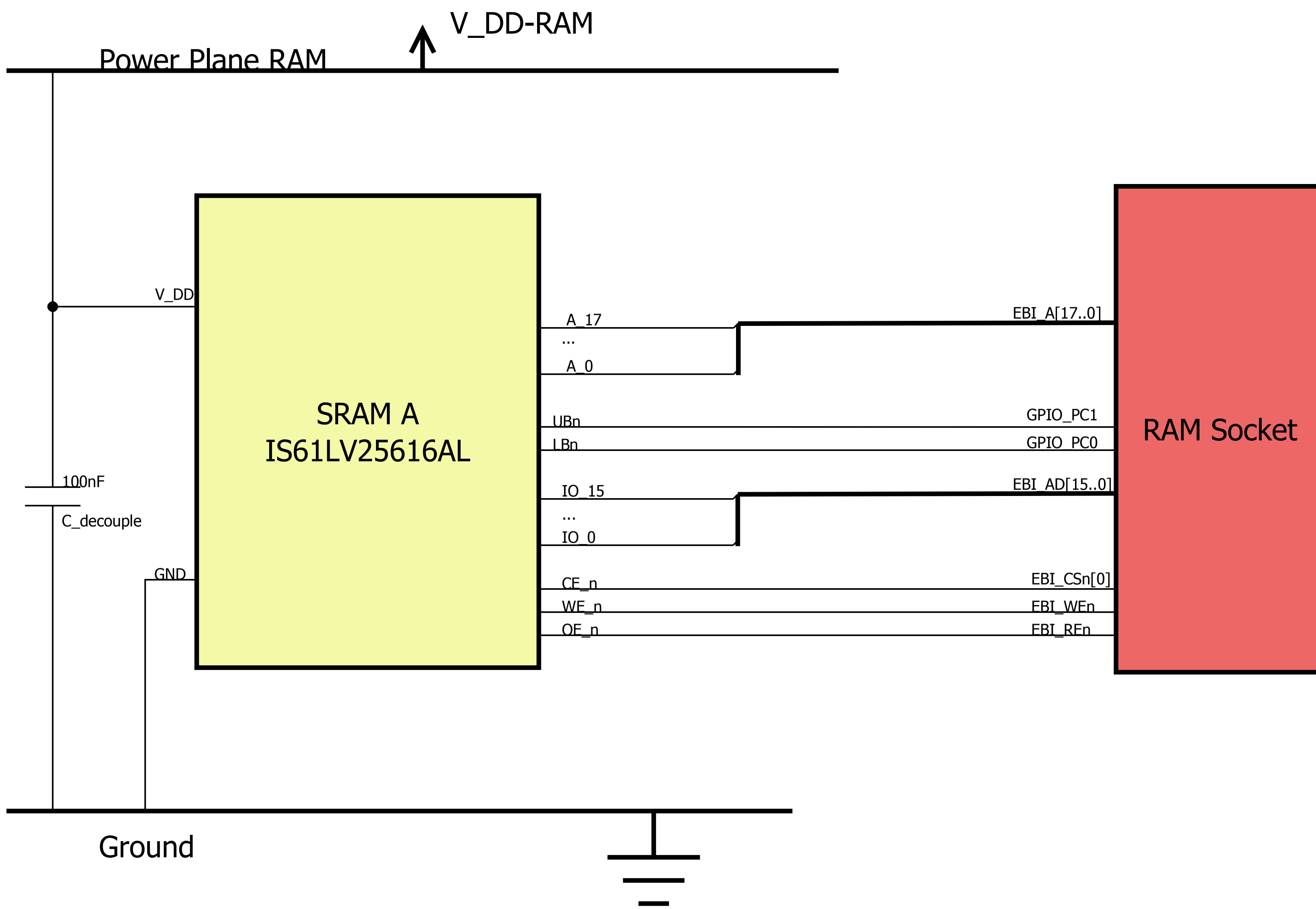


Figure B.8: Schematic of SRAM A.

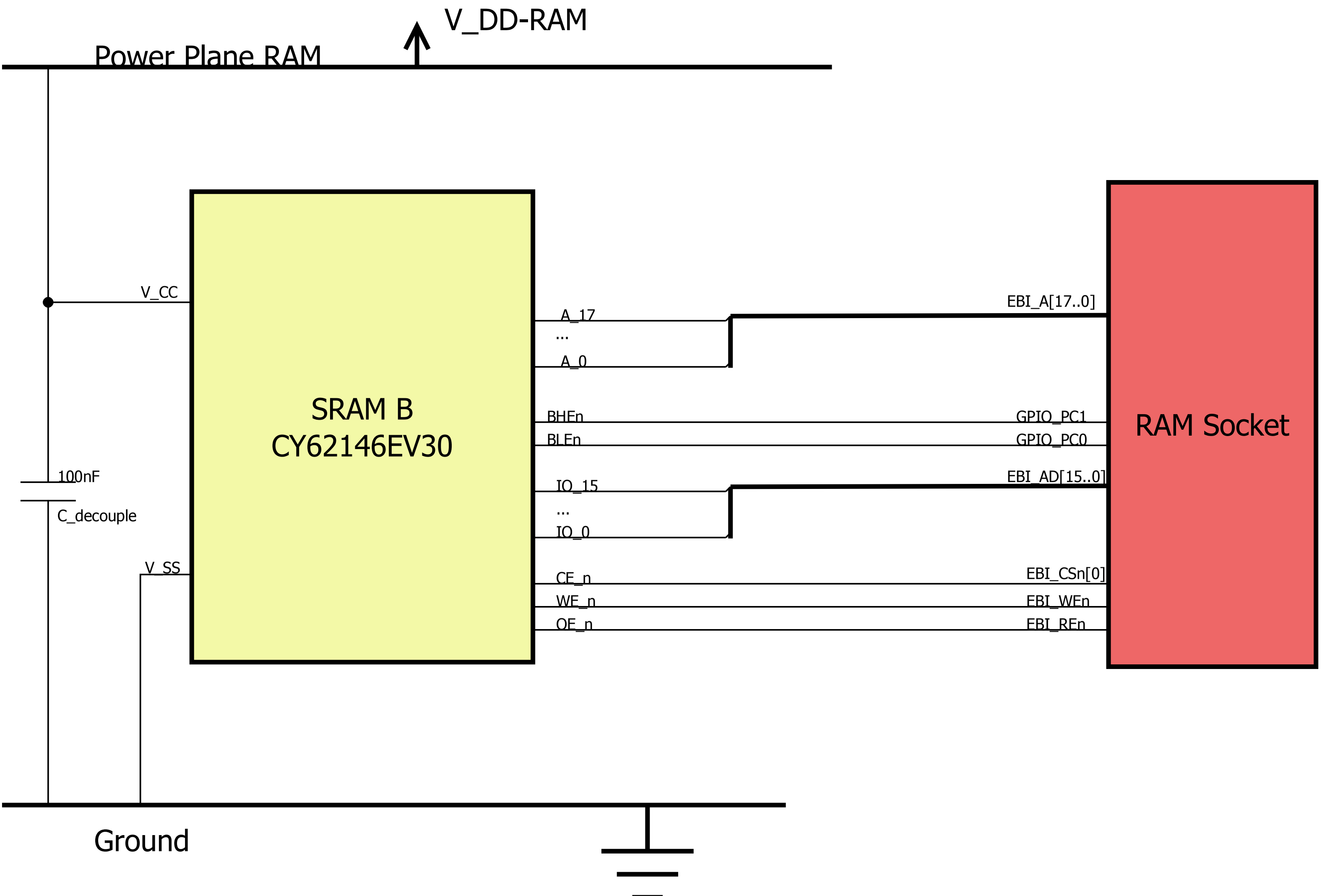


Figure B.9: Schematic of SRAM B.

C Appendix: C code

In this appendix, the software to run on the microcontroller during the measurements is included. Note that automatic line breaking is used to make sure that the code fits on the page.

```
1 /*********************************************
2  * @file
3  * @This file writes and reads an external RAM with a given duty
     cycle
4  * @details
5  *
6  * @author Magnus Moreau
7  * @contact magnus.moreau@gmail.com
8  * @version 0.90
9  *********************************************
10 * This software was written during my master thesis at NTNU.
11 *
12 *
13 * This software was based on the EBI application note example
     from
14 * Energy Micro AS. http://cdn.energymicro.com/dl/an/zip/
     an0034_efm32_ebi.zip
15 *********************************************/
16
17 #include <stdbool.h>
18 #include <stdlib.h>      /* srand, rand */
19 #include "em_device.h"
20 #include "em_chip.h"
21 #include "em_ebi.h"
22 #include "em_cmu.h"
23 #include "em_dbg.h"
24 #include "em_gpio.h"
25 #include "em_timer.h"
26 #include "em_emu.h"
27
28
29 #define EXT_RAM_BASE_ADDRESS     ((volatile uint16_t*) 0x80000000)
30 #define PERCENTAGE_WRITE              4 // Corresponds to 40%
31 #define PERCENTAGE_READ             10-PERCENTAGE_WRITE
32
33 #ifndef PERCENTAGE_WRITE
34   #define PERCENTAGE_WRITE                  1
35   #define PERCENTAGE_READ                   1
36 #endif
37
38 #define ACTIVE_COUNT   3
```

```
39 #define SLEEP_COUNT   12
40 #define WAKE_UP_COUNT 1 // Set to zero if the memory is volatile.
41
42
43
44 // Conditional defines:
45 //#define CHECK_MEMORY_CONTENT
46 #define GENERATE_RANDOM_DATA
47
48 #define NON_VOLATILE
49 // If the memory is defined as non volatile, the memory is powered
     down when the memory is sleeping
50
51 // Global variables *
52
53 //If the memory is not defined to generate random data, the test
    arrays are initialized.
54 #ifndef GENERATE_RANDOM_DATA
55 #define TEST_ARRAY_SIZE         16
56 uint16_t test_A[TEST_ARRAY_SIZE] = { 0x7BC1, 0x1EE2, 0x2E40, 0
    x9F96, 0xE93D, 0x7E11, 0x7393, 0x172A,
57                                    0xAE2D, 0x8A57, 0x1E03, 0xAC9C,
                                           0x9EB7, 0x6FAC, 0x45AF, 0
                                         x8E51,};
58 uint16_t test_B[TEST_ARRAY_SIZE] = { 0x30C8, 0x1C46, 0xA35C, 0
    xE411, 0xE5FB, 0xC119, 0x1A0A, 0x52EF,
59                                    0xF69F, 0x2445, 0xDF4F, 0x9B17,
                                           0xAD2B, 0x417B, 0xE66C, 0
                                         x3710 };
60 uint16_t answer_A[TEST_ARRAY_SIZE] = { 0x0000, 0x1111, 0x2222, 0
    x3333, 0x4444, 0x5555, 0x6666, 0x7777,
61                                      0x8888, 0x9999, 0xAAAA, 0
                                           xBBBB, 0xCCCC, 0xDDDD, 0
                                           xEEEE, 0xFFFF };
62 uint16_t answer_B[TEST_ARRAY_SIZE] = { 0x0000, 0x1111, 0x2222, 0
    x3333, 0x4444, 0x5555, 0x6666, 0x7777,
63                                      0x8888, 0x9999, 0xAAAA, 0
                                           xBBBB, 0xCCCC, 0xDDDD, 0
                                           xEEEE, 0xFFFF };
64 uint16_t ram_address[TEST_ARRAY_SIZE]= { 0xC8C1, 0x46E2, 0x5C40, 0
    xFB96, 0x193D, 0x1211, 0x0A93, 0xEF2A,
65                                         0x9F2D, 0x4557, 0x04F3, 0
                                              x179C, 0x2BB7, 0x7B6C,
                                              0x6DAF, 0x1051 };
66 #else
67 #define TEST_ARRAY_SIZE          32
68 uint16_t test_A[TEST_ARRAY_SIZE];
69 uint16_t test_B[TEST_ARRAY_SIZE];
70 uint16_t answer_A[TEST_ARRAY_SIZE];
```

```
71 uint16_t answer_B[TEST_ARRAY_SIZE];
72 uint16_t ram_address[TEST_ARRAY_SIZE];
73 #endif
74
75 uint32_t count = 0;
76 bool     active = true;
77 bool     table_A = true;
78
79 // Prototypes:
80
81 void write_external_ram(void);
82 void read_external_ram(void);
83 void check_memory_content(void);
84 void Power_Memory_Down(bool);
85
86
87 void TIMER0_IRQHandler(void)
88 {
89   /* Clear flag for TIMER0 overflow interrupt */
90   TIMER_IntClear(TIMER0, TIMER_IF_OF);
91
92   if(count<ACTIVE_COUNT){
93     active=true;
94   }else if(count<ACTIVE_COUNT+SLEEP_COUNT){
95     active=false;
96     Power_Memory_Down(true);
97   }else if(count<ACTIVE_COUNT+SLEEP_COUNT+WAKE_UP_COUNT){
98     active=false;
99     Power_Memory_Down(false);
100   }else{
101     // Reset counter
102     count=0;
103     active=true;
104   }
105   // Count number of overflows
106
107   count ++;
108
109 }
110
111 void TimerSetup(uint16_t top)
112 {
113   // Sets core clock to 1 MHz
114   CMU_HFRCOBandSet(cmuHFRCOBand_1MHz);
115   // Enable clock for TIMER0 module
116   CMU_ClockEnable(cmuClock_TIMER0, true);
117   // Select TIMER0 parameters
118   TIMER_Init_TypeDef timerInit =
119   {
```

```
120     .enable      = true,
121     .debugRun    = false,
122     .prescale    = timerPrescale64,
123     .clkSel      = timerClkSelHFPerClk,
124     .fallAction = timerInputActionNone,
125     .riseAction = timerInputActionNone,
126     .mode        = timerModeUp,
127     .dmaClrAct   = false,
128     .quadModeX4 = false,
129     .oneShot     = false,
130     .sync        = false,
131   };
132
133     // Enable overflow interrupt
134   TIMER_IntEnable(TIMER0, TIMER_IF_OF);
135
136   // Enable TIMER0 interrupt vector in NVIC
137   NVIC_EnableIRQ(TIMER0_IRQn);
138
139   // Set TIMER Top value
140   TIMER_TopSet(TIMER0, top);
141
142   // Configure TIMER
143   TIMER_Init(TIMER0, &timerInit);
144
145    // Enable clock for TIMER1 module
146   CMU_ClockEnable(cmuClock_TIMER1, true);
147   // Select TIMER1 parameters
148   TIMER_Init_TypeDef timerInit1 =
149   {
150     .enable      = false,
151     .debugRun    = false,
152     .prescale    = timerPrescale1,
153     .clkSel      = timerClkSelHFPerClk,
154     .fallAction = timerInputActionNone,
155     .riseAction = timerInputActionNone,
156     .mode        = timerModeUp,
157     .dmaClrAct   = false,
158     .quadModeX4 = false,
159     .oneShot     = 1,
160     .sync        = false,
161   };
162
163
164   // Set TIMER Top value
165   TIMER_TopSet(TIMER1, 0xffff);
166
167   // Configure TIMER
168   TIMER_Init(TIMER1, &timerInit1);
```

```
169
170 }
171
172 void my_EBI_setup(void){
173
174   // Configure Pins for EBI
175
176   /* Pin PA0 is configured to Push-pull */
177   GPIO->P[0].MODEL = (GPIO->P[0].MODEL & ~_GPIO_P_MODEL_MODE0_MASK
        ) | GPIO_P_MODEL_MODE0_PUSHPULL;
178   /* Pin PA1 is configured to Push-pull */
179   GPIO->P[0].MODEL = (GPIO->P[0].MODEL & ~_GPIO_P_MODEL_MODE1_MASK
        ) | GPIO_P_MODEL_MODE1_PUSHPULL;
180   /* Pin PA2 is configured to Push-pull */
181   GPIO->P[0].MODEL = (GPIO->P[0].MODEL & ~_GPIO_P_MODEL_MODE2_MASK
        ) | GPIO_P_MODEL_MODE2_PUSHPULL;
182   /* Pin PA3 is configured to Push-pull */
183   GPIO->P[0].MODEL = (GPIO->P[0].MODEL & ~_GPIO_P_MODEL_MODE3_MASK
        ) | GPIO_P_MODEL_MODE3_PUSHPULL;
184   /* Pin PA4 is configured to Push-pull */
185   GPIO->P[0].MODEL = (GPIO->P[0].MODEL & ~_GPIO_P_MODEL_MODE4_MASK
        ) | GPIO_P_MODEL_MODE4_PUSHPULL;
186   /* Pin PA5 is configured to Push-pull */
187   GPIO->P[0].MODEL = (GPIO->P[0].MODEL & ~_GPIO_P_MODEL_MODE5_MASK
        ) | GPIO_P_MODEL_MODE5_PUSHPULL;
188   /* Pin PA6 is configured to Push-pull */
189   GPIO->P[0].MODEL = (GPIO->P[0].MODEL & ~_GPIO_P_MODEL_MODE6_MASK
        ) | GPIO_P_MODEL_MODE6_PUSHPULL;
190   /* Pin PA12 is configured to Push-pull */
191   GPIO->P[0].MODEH = (GPIO->P[0].MODEH & ~
        _GPIO_P_MODEH_MODE12_MASK) | GPIO_P_MODEH_MODE12_PUSHPULL;
192   /* Pin PA13 is configured to Push-pull */
193   GPIO->P[0].MODEH = (GPIO->P[0].MODEH & ~
        _GPIO_P_MODEH_MODE13_MASK) | GPIO_P_MODEH_MODE13_PUSHPULL;
194   /* Pin PA14 is configured to Push-pull */
195   GPIO->P[0].MODEH = (GPIO->P[0].MODEH & ~
        _GPIO_P_MODEH_MODE14_MASK) | GPIO_P_MODEH_MODE14_PUSHPULL;
196   /* Pin PA15 is configured to Push-pull */
197   GPIO->P[0].MODEH = (GPIO->P[0].MODEH & ~
        _GPIO_P_MODEH_MODE15_MASK) | GPIO_P_MODEH_MODE15_PUSHPULL;
198   /* Pin PB0 is configured to Push-pull */
199   GPIO->P[1].MODEL = (GPIO->P[1].MODEL & ~_GPIO_P_MODEL_MODE0_MASK
        ) | GPIO_P_MODEL_MODE0_PUSHPULL;
200   /* Pin PB1 is configured to Push-pull */
201   GPIO->P[1].MODEL = (GPIO->P[1].MODEL & ~_GPIO_P_MODEL_MODE1_MASK
        ) | GPIO_P_MODEL_MODE1_PUSHPULL;
202   /* Pin PB2 is configured to Push-pull */
203   GPIO->P[1].MODEL = (GPIO->P[1].MODEL & ~_GPIO_P_MODEL_MODE2_MASK
        ) | GPIO_P_MODEL_MODE2_PUSHPULL;
```

```
204    /* Pin PB3 is configured to Push-pull */
205    GPIO->P[1].MODEL = (GPIO->P[1].MODEL & ~_GPIO_P_MODEL_MODE3_MASK
           ) | GPIO_P_MODEL_MODE3_PUSHPULL;
206    /* Pin PB4 is configured to Push-pull */
207    GPIO->P[1].MODEL = (GPIO->P[1].MODEL & ~_GPIO_P_MODEL_MODE4_MASK
           ) | GPIO_P_MODEL_MODE4_PUSHPULL;
208    /* Pin PB5 is configured to Push-pull */
209    GPIO->P[1].MODEL = (GPIO->P[1].MODEL & ~_GPIO_P_MODEL_MODE5_MASK
           ) | GPIO_P_MODEL_MODE5_PUSHPULL;
210    /* Pin PB6 is configured to Push-pull */
211    GPIO->P[1].MODEL = (GPIO->P[1].MODEL & ~_GPIO_P_MODEL_MODE6_MASK
           ) | GPIO_P_MODEL_MODE6_PUSHPULL;
212    /* Pin PB9 is configured to Push-pull */
213    GPIO->P[1].MODEH = (GPIO->P[1].MODEH & ~_GPIO_P_MODEH_MODE9_MASK
           ) | GPIO_P_MODEH_MODE9_PUSHPULL;
214    /* Pin PB10 is configured to Push-pull */
215    GPIO->P[1].MODEH = (GPIO->P[1].MODEH & ~
           _GPIO_P_MODEH_MODE10_MASK) | GPIO_P_MODEH_MODE10_PUSHPULL;
216    /* Pin PC6 is configured to Push-pull */
217    GPIO->P[2].MODEL = (GPIO->P[2].MODEL & ~_GPIO_P_MODEL_MODE6_MASK
           ) | GPIO_P_MODEL_MODE6_PUSHPULL;
218    /* Pin PC7 is configured to Push-pull */
219    GPIO->P[2].MODEL = (GPIO->P[2].MODEL & ~_GPIO_P_MODEL_MODE7_MASK
           ) | GPIO_P_MODEL_MODE7_PUSHPULL;
220    /* Pin PC8 is configured to Push-pull */
221    GPIO->P[2].MODEH = (GPIO->P[2].MODEH & ~_GPIO_P_MODEH_MODE8_MASK
           ) | GPIO_P_MODEH_MODE8_PUSHPULL;
222    /* Pin PC9 is configured to Push-pull */
223    GPIO->P[2].MODEH = (GPIO->P[2].MODEH & ~_GPIO_P_MODEH_MODE9_MASK
           ) | GPIO_P_MODEH_MODE9_PUSHPULL;
224    /* Pin PC10 is configured to Push-pull */
225    GPIO->P[2].MODEH = (GPIO->P[2].MODEH & ~
           _GPIO_P_MODEH_MODE10_MASK) | GPIO_P_MODEH_MODE10_PUSHPULL;
226    /* Pin PD9 is configured to Push-pull */
227    GPIO->P[3].MODEH = (GPIO->P[3].MODEH & ~_GPIO_P_MODEH_MODE9_MASK
           ) | GPIO_P_MODEH_MODE9_PUSHPULL;
228    /* Pin PE0 is configured to Push-pull */
229    GPIO->P[4].MODEL = (GPIO->P[4].MODEL & ~_GPIO_P_MODEL_MODE0_MASK
           ) | GPIO_P_MODEL_MODE0_PUSHPULL;
230    /* Pin PE1 is configured to Push-pull */
231    GPIO->P[4].MODEL = (GPIO->P[4].MODEL & ~_GPIO_P_MODEL_MODE1_MASK
           ) | GPIO_P_MODEL_MODE1_PUSHPULL;
232    /* Pin PE4 is configured to Push-pull */
233    GPIO->P[4].MODEL = (GPIO->P[4].MODEL & ~_GPIO_P_MODEL_MODE4_MASK
           ) | GPIO_P_MODEL_MODE4_PUSHPULL;
234    /* Pin PE5 is configured to Push-pull */
235    GPIO->P[4].MODEL = (GPIO->P[4].MODEL & ~_GPIO_P_MODEL_MODE5_MASK
           ) | GPIO_P_MODEL_MODE5_PUSHPULL;
236    /* Pin PE6 is configured to Push-pull */
```

```
237   GPIO->P[4].MODEL = (GPIO->P[4].MODEL & ~_GPIO_P_MODEL_MODE6_MASK
          ) | GPIO_P_MODEL_MODE6_PUSHPULL;
238   /* Pin PE7 is configured to Push-pull */
239   GPIO->P[4].MODEL = (GPIO->P[4].MODEL & ~_GPIO_P_MODEL_MODE7_MASK
          ) | GPIO_P_MODEL_MODE7_PUSHPULL;
240   /* Pin PE8 is configured to Push-pull */
241   GPIO->P[4].MODEH = (GPIO->P[4].MODEH & ~_GPIO_P_MODEH_MODE8_MASK
          ) | GPIO_P_MODEH_MODE8_PUSHPULL;
242   /* Pin PE9 is configured to Push-pull */
243   GPIO->P[4].MODEH = (GPIO->P[4].MODEH & ~_GPIO_P_MODEH_MODE9_MASK
          ) | GPIO_P_MODEH_MODE9_PUSHPULL;
244   /* Pin PE10 is configured to Push-pull */
245   GPIO->P[4].MODEH = (GPIO->P[4].MODEH & ~
          _GPIO_P_MODEH_MODE10_MASK) | GPIO_P_MODEH_MODE10_PUSHPULL;
246   /* Pin PE11 is configured to Push-pull */
247   GPIO->P[4].MODEH = (GPIO->P[4].MODEH & ~
          _GPIO_P_MODEH_MODE11_MASK) | GPIO_P_MODEH_MODE11_PUSHPULL;
248   /* Pin PE12 is configured to Push-pull */
249   GPIO->P[4].MODEH = (GPIO->P[4].MODEH & ~
          _GPIO_P_MODEH_MODE12_MASK) | GPIO_P_MODEH_MODE12_PUSHPULL;
250   /* Pin PE13 is configured to Push-pull */
251   GPIO->P[4].MODEH = (GPIO->P[4].MODEH & ~
          _GPIO_P_MODEH_MODE13_MASK) | GPIO_P_MODEH_MODE13_PUSHPULL;
252   /* Pin PE14 is configured to Push-pull */
253   GPIO->P[4].MODEH = (GPIO->P[4].MODEH & ~
          _GPIO_P_MODEH_MODE14_MASK) | GPIO_P_MODEH_MODE14_PUSHPULL;
254   /* Pin PE15 is configured to Push-pull */
255   GPIO->P[4].MODEH = (GPIO->P[4].MODEH & ~
          _GPIO_P_MODEH_MODE15_MASK) | GPIO_P_MODEH_MODE15_PUSHPULL;
256   /* Pin PF8 is configured to Push-pull */
257   GPIO->P[5].MODEH = (GPIO->P[5].MODEH & ~_GPIO_P_MODEH_MODE8_MASK
          ) | GPIO_P_MODEH_MODE8_PUSHPULL;
258   /* Pin PF9 is configured to Push-pull */
259   GPIO->P[5].MODEH = (GPIO->P[5].MODEH & ~_GPIO_P_MODEH_MODE9_MASK
          ) | GPIO_P_MODEH_MODE9_PUSHPULL;
260
261   /* Enable clock for EBI */
262   CMU_ClockEnable(cmuClock_EBI, true);
263   /* Module EBI is configured to location 1 */
264   EBI->ROUTE = (EBI->ROUTE & ~_EBI_ROUTE_LOCATION_MASK) |
          EBI_ROUTE_LOCATION_LOC1;
265   /* EBI I/O routing */
266   EBI->ROUTE |= EBI_ROUTE_APEN_A23 | EBI_ROUTE_CS0PEN |
          EBI_ROUTE_EBIPEN;
267
268   EBI_Init_TypeDef ebiConfig = EBI_INIT_DEFAULT;
269
270   //Standard setup from the ebi setup function with some changes:
271   //Prefetch disabled and all hold times set to 0.
```

```
272
273   ebiConfig.banks         = EBI_BANK0;
274   ebiConfig.csLines       = EBI_CS0;
275   ebiConfig.mode          = ebiModeD16;
276   ebiConfig.alePolarity   = ebiActiveHigh;
277   ebiConfig.blEnable      = true;
278   ebiConfig.noIdle        = true;
279   ebiConfig.ardyEnable    = false;
280   ebiConfig.addrHalfALE   = true;
281   ebiConfig.readPrefetch  = false;
282   ebiConfig.aLow          = ebiALowA0;
283   ebiConfig.aHigh         = ebiAHighA23;
284   ebiConfig.location      = ebiLocation1;
285
286   // Address Setup and hold time
287   ebiConfig.addrHoldCycles  = 0;
288   ebiConfig.addrSetupCycles = 0;
289
290   // Read cycle times
291   ebiConfig.readStrobeCycles = 0;
292   ebiConfig.readHoldCycles   = 0;
293   ebiConfig.readSetupCycles  = 0;
294
295   // Write cycle times
296   ebiConfig.writeStrobeCycles = 0;
297   ebiConfig.writeHoldCycles   = 0;
298   ebiConfig.writeSetupCycles  = 0;
299
300   EBI_Init(&ebiConfig);
301
302 }
303 void my_GPIO_setup(void){
304   // Enable clock for GPIO module
305   CMU_ClockEnable(cmuClock_GPIO, true);
306     // Configure  LEDS PD0-1 as push pull output
307   GPIO_PinModeSet(gpioPortD, 0, gpioModePushPullDrive, 0);
308   GPIO_PinModeSet(gpioPortD, 1, gpioModePushPullDrive, 0);
309   GPIO_PinModeSet(gpioPortD, 2, gpioModePushPullDrive, 0);
310   GPIO_PinModeSet(gpioPortD, 3, gpioModePushPullDrive, 0);
311
312     // Configure PD4-5 as input with filter
313   GPIO_PinModeSet(gpioPortD, 4, gpioModeInput, 0);
314   GPIO_PinModeSet(gpioPortD, 5, gpioModeInput, 0);
315
316    // Configure  PowerDown PD6 as push pull output
317   GPIO_PinModeSet(gpioPortD, 6, gpioModePushPullDrive, 0);
318
319 }
320
```

```
321 void random_data_init(void){
322     // Fill the arrays with random numbers.
323     for (uint32_t i=0 ; i<TEST_ARRAY_SIZE; i++)
324     {
325       test_A[i]    = rand();
326       test_B[i]    = rand();
327       answer_A[i] = rand();
328       answer_B[i] = rand();
329       ram_address[i] = rand();
330     }
331 }
332
333
334 void write_external_ram(void){
335     /* Write external RAM */
336   for (uint32_t j=0; j<PERCENTAGE_WRITE; j++){
337     // Read timer value to estimate length of one active cycle.
338
339     if(!active){ // if the active period is over stop this
          function and return to main
340       return;
341     }
342     //Alternating by writing the A and B test array in order to
          make sure not to write the same data all the time.
343     if(table_A){
344       for (uint32_t i=0 ; i<TEST_ARRAY_SIZE ; i++)
345       {
346
347         /* Write data in the External SRAM */
348         *(uint16_t*)(EXT_RAM_BASE_ADDRESS + ram_address[i]) =
              test_A[i];
349
350       }// end for i
351     }else{
352        for (uint32_t i=0 ; i<TEST_ARRAY_SIZE ; i++)
353       {
354         /* Write data in the External SRAM */
355         *(uint16_t*)(EXT_RAM_BASE_ADDRESS + ram_address[i]) =
              test_B[i];
356       }// end for i
357     } // end else
358     table_A = !table_A;
359   } // end for j
360 }
361
362 void read_external_ram(void){
363   /* Read external RAM*/
364   for (uint32_t j=0; j<PERCENTAGE_READ; j++){
365     if(!active){ // if the active period is over stop this
```

```
            function and return to main
366        return;
367      }
368      //Using the same algorithm for reading as writing. Not really
            neccessary, but done in order to make sure the algorithms
            for writing and reading are aligned.
369      if(table_A){
370      for (uint32_t i=0 ; i<TEST_ARRAY_SIZE; i++)
371      {
372        /* Read data from External SRAM */
373        answer_A[i] = *(uint16_t*)(EXT_RAM_BASE_ADDRESS +
              ram_address[i]);
374      }
375      }else{
376        for (uint32_t i=0 ; i<TEST_ARRAY_SIZE ; i++)
377        {
378          /* Write data in the External SRAM */
379          answer_B[i] = *(uint16_t*)(EXT_RAM_BASE_ADDRESS +
                ram_address[i]);
380        }// end for i
381      } // end else
382      table_A = !table_A;
383    }
384  }
385
386  void check_memory_content(void){
387      bool error = false;
388      // Check for any differences between the arrays.
389      for (uint32_t i = 0; i < TEST_ARRAY_SIZE; i++)
390      {
391        if (test_A[i] != answer_A[i])
392        {
393          error = true;
394        }
395      }
396      if (error)
397      {
398        /* Write and Read operation  FAILED */
399        while (1);
400      }
401      else
402      {
403        /* Write and Read operation  SUCCESS */
404        while (1);
405      }
406  }
407
408  void Power_Memory_Down(bool Power_Down){
409  #if defined( NON_VOLATILE ) // Only power down if the memory is
```

```
     defined as non-volatile, if the memory is volatile this
     function does nothing.
410   if (Power_Down){
411     GPIO_PinOutSet(gpioPortD, 6);
412   }else{
413     GPIO_PinOutClear(gpioPortD, 6);
414   }
415 #endif
416 }
417 //************************************
418 // Main fuction:
419 //************************************
420 int main(void)
421 {
422
423   // Chip revision alignment and errata fixes
424   CHIP_Init();
425   // Setup timer and clocks.
426   TimerSetup(100000/64);
427   my_GPIO_setup(); // Setup GPIO
428   my_EBI_setup(); // Setup EBI
429
430 #if defined( GENERATE_RANDOM_DATA )
431   random_data_init(); //Initialize arrays with random data
432 #endif
433
434   //Main while loop, toggling between active and !active
435   while(1){
436     if(active){
437
438       write_external_ram();
439       read_external_ram();
440
441     }else{//if(!active)
442
443       EMU_EnterEM1(); // Go to sleep mode
444
445     }
446
447   #if defined( CHECK_MEMORY_CONTENT )
448     check_memory_content();
449   #endif
450   } // end while(1)
451 } // end main function
```

D Appendix: Matlab Scripts for Case Studies

In this appendix the Matlab script developed to normalize and calculate the power optimal power consumption for the selected memories. After the power is calculated it is plotted and saved.

```
clear all
close all
clc

% Chips:
% FeRAM A= MB85R256F
% FeRAM B = FM22LD16
% MRAM A = MR256A08B
% MRAM B = MR2A16A
% PCRAM = NP8P128A13TSM60E
% SRAM A = IS61LV25616AL
% SRAM B = CY62146EV30

% Save data to pdf:
save = true;

% Text scaling:
d=2;
scale=1.5;

groups={'FeRAM A', 'FeRAM B', 'MRAM A' ,'MRAM B' ,'PCRAM' , ...
    'SRAM A','SRAM B'};

f=32e6;% Frequency for normalization
Ws=16;% Wordsize for normalization
percentage_write=0.4;
percentage_read=1-percentage_write;

%Data sheet data:
supply_voltage_max=[3.6, 3.6, 3.6, 3.6, 3.6, 3.6, 3.6];
active_current_write_max=[10, 12, 65, 155, 42, 100, 20 ]*1e-3;
active_current_read_max=[10, 12, 30, 80, 42, 100, 20 ]*1e-3;
word_size_write=[8, 16, 8, 16, 8*64, 16, 16];
word_size_read=[8, 16, 8, 16, 16, 16, 16];
min_write_cycle_time=[150, 110, 35, 35, 120000, 10, 45]*1e-9;
min_read_cycle_time=[150, 110, 35, 35, 200, 10, 45]*1e-9;
standby_current_max=[50, 270, 6000, 12000, 160, 15000, 7]*1e-6;

volatile=[false,false,false,false,false,true,true];
memory_size=[256*2^10, 4*2^20, 32*8*2^10, 4*2^20, 128*2^20,
    256*16*2^10, 256*16*2^10];
```

```
41 start_up_time_min=[50, 600, 2000, 2000, 100, 0,0]*1e-6;
42
43 tau = active_current_write_max ./standby_current_max.*
       start_up_time_min;
44 %Output
45 E_write_per_bit= supply_voltage_max .* active_current_write_max .*
        ...
46      min_write_cycle_time./word_size_write;
47 E_read_per_bit= supply_voltage_max .* active_current_read_max .*
       ...
48      min_read_cycle_time./word_size_read;
49
50 figure;
51 bar(E_write_per_bit*1e12)
52 ylabel('E_{write/bit} [pJ]')
53 ylim([1 1e5])
54 title('Estimated write energy per bit of the selected memory chips
       ')
55 set(gca,'xticklabel',groups,'fontsize',10)
56 logbar()
57 if(save)
58     saveas(gcf,'chips_E_write','pdf')
59 end
60
61 figure;
62 bar(E_read_per_bit*1e12)
63 ylabel('E_{read/bit} [pJ]')
64 title('Estimated read energy per bit of the selected memory chips'
       )
65 set(gca,'xticklabel',groups,'fontsize',10)
66 ylim([1 1e5])
67 logbar()
68 if(save)
69     saveas(gcf,'chips_E_read','pdf')
70 end
71
72
73
74 % %Active
75
76 active_power_write=Ws*f*E_write_per_bit;
77 active_power_read=Ws*f*E_read_per_bit;
78
79 figure;
80 bar([active_power_write' active_power_read']*1e3)
81 barnumber([active_power_write' active_power_read']*1e3,length(
       groups))
82 ylabel('P [mW]')
83 ylim([1 1e5])
```

```
84 legend('P_{write}','P_{read}','Location','NorthWest')
85 title('Write and read power of selected memories, normalized for
       frequency and number of I/Os')
86 set(gca,'xticklabel',groups,'fontsize',10)
87 logbar()
88 if(save)
89     saveas(gcf,'chips_P','pdf')
90 end
91
92
93
94 active_power=active_power_write.*percentage_write ...
95     + active_power_read.*percentage_read;
96
97 duty_lim=6;
98 num_points=1000;
99 power=zeros(7,num_points);
100
101 for j=1:2:7
102 active_time=10^-j;
103
104
105 title_str=sprintf('Power vs Duty cycle for selected memories.\n
       Active time = %g s',active_time(1));
106
107 active_time=active_time*ones(1,num_points);
108 passive_time=logspace(log10(active_time(1))-2,log10(active_time(1)
       )+duty_lim,num_points);
109 passive_energy=ones(length(groups),length(passive_time));
110 for i=1:length(groups)
111     if(volatile(i))
112         passive_energy(i,:)=supply_voltage_max(i).*
                standby_current_max(i).*passive_energy(i,:).*
                passive_time;
113     else
114         passive_energy(i,:)=min(supply_voltage_max(i).*
                standby_current_max(i).*passive_energy(i,:).*
                passive_time...
115             ,supply_voltage_max(i).*active_current_write_max(i)...
116             .*start_up_time_min(i));
117     end
118 end
119 active_energy=repmat(active_power.*active_time(1),num_points,1)';
120
121 duty_cycle=active_time./(active_time+passive_time);
122
123
124 for i=1:length(groups)
125     power(i,:)=(active_energy(i,:)+passive_energy(i,:))./(
```

```
            active_time(1,:)+passive_time(1,:));
126 end
127
128 figure;
129 loglog(duty_cycle,power(1,:),'k')
130 xlim([10^(-duty_lim) 1])
131 ylim([1e-6 1e2])
132 hold on
133 loglog(duty_cycle,power(2,:),'r')
134 loglog(duty_cycle,power(3,:),'g')
135 loglog(duty_cycle,power(4,:),'Color',[1,0.5,0])
136 loglog(duty_cycle,power(5,:),'c')
137 loglog(duty_cycle,power(6,:),'b')
138 loglog(duty_cycle,power(7,:),'m')
139 ylabel('P [W]')
140 xlabel('Duty cycle')
141 title(title_str)
142 legend(groups,'Location','NorthWest')
143
144 [xout,yout] = intersections(duty_cycle,power(7,:),duty_cycle,power
        (1,:),1);
145 loglog(xout,yout,'k.','markersize',10)
146 text(xout,yout*scale,char(vpa(xout,d)),'fontsize',12,'
        horizontalAlignment','left','color','k')
147
148 [xout,yout] = intersections(duty_cycle,power(6,:),duty_cycle,power
        (1,:),1);
149 loglog(xout,yout,'k.','markersize',10)
150 text(xout,yout*scale,char(vpa(xout,d)),'fontsize',12,'
        horizontalAlignment','center','color','k')
151
152
153 [xout,yout] = intersections(duty_cycle,power(7,:),duty_cycle,power
        (2,:),1);
154 loglog(xout,yout,'r.','markersize',10)
155 text(xout,yout/scale,char(vpa(xout,d)),'fontsize',12,'
        horizontalAlignment','left','color','r')
156
157 [xout,yout] = intersections(duty_cycle,power(6,:),duty_cycle,power
        (2,:),1);
158 loglog(xout,yout,'r.','markersize',10)
159 text(xout,yout/scale,char(vpa(xout,d)),'fontsize',12,'
        horizontalAlignment','left','color','r')
160
161
162 [xout,yout] = intersections(duty_cycle,power(7,:),duty_cycle,power
        (3,:),1);
163 loglog(xout,yout,'g.','markersize',10)
164 text(xout,yout/scale,char(vpa(xout,d)),'fontsize',12,'
```

```
    horizontalAlignment','right','color','g')
165
166 [xout,yout] = intersections(duty_cycle,power(6,:),duty_cycle,power
    (3,:),1);
167 loglog(xout,yout,'g.','markersize',10)
168 text(xout,yout/scale,char(vpa(xout,d)),'fontsize',12,'
    horizontalAlignment','right','color','g')
169
170
171 [xout,yout] = intersections(duty_cycle,power(7,:),duty_cycle,power
    (4,:),1);
172 loglog(xout,yout,'.','Color',[1,0.5,0],'markersize',10)
173 text(xout,yout*scale,char(vpa(xout,d)),'fontsize',12,'
    horizontalAlignment','right','Color',[1,0.5,0])
174
175 [xout,yout] = intersections(duty_cycle,power(6,:),duty_cycle,power
    (4,:),1);
176 loglog(xout,yout,'.','Color',[1,0.5,0],'markersize',10)
177 text(xout,yout*scale,char(vpa(xout,d)),'fontsize',12,'
    horizontalAlignment','right','Color',[1,0.5,0])
178
179 [xout,yout] = intersections(duty_cycle,power(7,:),duty_cycle,power
    (5,:),1);
180 loglog(xout,yout,'c.','markersize',10)
181 if(active_time(1)==0.0001)
182     text(xout,yout*scale,char(vpa(xout,d)),'fontsize',12,'
        horizontalAlignment','left','color','c')
183 else
184     text(xout,yout/scale,char(vpa(xout,d)),'fontsize',12,'
        horizontalAlignment','left','color','c')
185 end
186 [xout,yout] = intersections(duty_cycle,power(6,:),duty_cycle,power
    (5,:),1);
187 loglog(xout,yout,'c.','markersize',10)
188 text(xout,yout/scale,char(vpa(xout,d)),'fontsize',12,'
    horizontalAlignment','left','color','c')
189
190
191
192 hold off
193
194 if(save)
195     filename=sprintf('chips_duty_active=%g',active_time(1));
196     filename = strrep(filename, '.', ',');
197     saveas(gcf,filename,'pdf')
198 end
199 end
200
201 if(save)
```

```
202     close all
203 end
```