AN EXAMINATION OF POST-CMOS COMPUTING TECHNIQUES
USING STEEP SLOPE DEVICE-BASED ARCHITECTURES

A Dissertation in
Computer Science and Engineering
by
Karthik Swaminathan

© 2014 Karthik Swaminathan

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

December 2014
The dissertation of Karthik Swaminathan was reviewed and approved* by the following:

Vijaykrishnan Narayanan  
Professor of Computer Science and Engineering  
Advisor  
Chair of Committee

Mahmut Kandemir  
Professor of Computer Science and Engineering

Anand Sivasubramaniam  
Professor of Computer Science and Engineering

Jack Sampson  
Assistant Professor of Computer Science and Engineering

Jeya Chandra  
Professor of Industrial Engineering

Lee Coraor  
Professor of Graduate Program Chair  
Computer Science and Engineering

*Signatures are on file in the Graduate School.
Abstract

As technology scaling results in smaller and smaller transistor sizes, existing CMOS technology starts to encounter certain fundamental physical limitations. These limitations can potentially result in drastic reduction in performance and energy efficiencies of these transistors, which could make it untenable to continue scaling them. This in turn, would severely limit the benefits of further innovations in processor design and architecture. In this scenario, there have been several alternatives proposed at the device level. However, each of these devices comes with varying energy-performance tradeoffs and optimal design points in comparison to CMOS technology. Hence, there is a need for a rethink of the complete design hierarchy, ranging from a simple circuit to entire processor platforms. In my work, I attempt to answer the question, Does a better device make a better processor?, by examining the impact of varying device parameters at the architecture-level. I carry out a comprehensive exploration of different design points, at the micro-architecture and architecture levels of abstraction and examine the feasibility of device-heterogeneous architectures for achieving greater regions of optimality. My work also extends to gauging the impact of using these emerging technologies on non-traditional architectures, including domain specific accelerators and 3D-stacked processors.
# Table of Contents

List of Tables ........................................ vii
List of Figures ........................................ viii
Acknowledgments ...................................... xi

Chapter 1. Introduction ............................... 1

Chapter 2. The landscape of emerging technologies - An overview .......................... 6
  2.1 Basic background on Steep Slope devices ............................................. 6
    2.1.1 The Manufacturing Barrier: Integrating TFET and FinFET Technologies .... 9

Chapter 3. Design of Heterogeneous CMOS-TFET multicores - From device to processor 10
  3.1 Introduction ...................................... 10
  3.2 Device-to-processor abstraction model .................................................. 10
  3.3 Variation in steep-slope multicores ...................................................... 15
    3.3.1 Observing device variations at the architecture level ..................... 16
    3.3.2 Variation mitigation using voltage scaling .................................... 16
  3.4 Impact of technology scaling ............................................................... 17

Chapter 4. Architectural exploration of device-heterogeneous multicores .................. 20
  4.1 Comparison of CMOS/TFET cores from the perspectives of dark and dim silicon .... 21
  4.2 Energy optimizations on CMOS-TFET multicores ..................................... 22
    4.2.1 Voltage scaling for Multithreaded Applications .............................. 22
    4.2.2 EDP-Aware DVFS ........................................................................ 23
    4.2.3 Barrier-Aware DVFS .................................................................. 25
    4.2.4 Thread Migration ........................................................................ 26
    4.2.5 Simulation Framework and Infrastructure ...................................... 27
    4.2.6 Experimental Results ................................................................... 28
    4.2.7 Energy Improvements .................................................................... 29
    4.2.8 Performance Degradation .............................................................. 31
    4.2.9 Overall Impact on EDP .................................................................. 32
  4.3 Performance Enhancement under Power Constraints for Heterogeneous CMOS-
      TFET Multicores ................................................................................... 32
    4.3.1 Motivation ..................................................................................... 32
    4.3.2 Serving a Diversity of Applications ............................................... 34
    4.3.3 Exploiting Device-Level Heterogeneity ........................................... 34
    4.3.4 Porting Applications to Heterogeneous Multicores ........................... 36
    4.3.5 Thread-to-Core Mapping ................................................................. 36
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.3.6 Work Partitioning</td>
<td>36</td>
</tr>
<tr>
<td>4.3.7 Power Partitioning</td>
<td>37</td>
</tr>
<tr>
<td>4.3.8 Overview of Evaluated Schemes</td>
<td>38</td>
</tr>
<tr>
<td>4.3.9 System Parameters and Simulation Infrastructure</td>
<td>39</td>
</tr>
<tr>
<td>4.3.10 Simulation Infrastructure</td>
<td>39</td>
</tr>
<tr>
<td>4.3.11 Dynamic Work and Power Partitioning Implementation</td>
<td>40</td>
</tr>
<tr>
<td>4.3.12 Benchmarks</td>
<td>41</td>
</tr>
<tr>
<td>4.3.13 Experimental Results</td>
<td>42</td>
</tr>
<tr>
<td>4.3.15 Hetero-Auto vs. Hetero-Manual-Best</td>
<td>43</td>
</tr>
<tr>
<td>4.3.16 Runtime Characteristics</td>
<td>45</td>
</tr>
<tr>
<td>4.3.17 Sensitivity Analysis</td>
<td>46</td>
</tr>
<tr>
<td>4.3.18 Multiple Multi-threaded Applications Sharing a Heterogeneous Multi-core</td>
<td>47</td>
</tr>
</tbody>
</table>

Chapter 5. Co-design of device and architecture in CMOS-TFET processors

5.1 Introduction                                                     51
5.2 Architectural considerations                                     53
  5.2.1 Thermal constraints based microarchitecture design           54
  5.2.2 Frequency-complexity Tradeoffs                               55
5.3 Architecture Design Details                                       57
  5.3.1 Static mapping of applications                               57
  5.3.2 Slack-based dynamic mapping                                  58
5.4 Simulation Infrastructure and Results                             59
  5.4.1 Simulation tools                                             59
  5.4.2 Results: Static mapping of applications                      59
  5.4.3 Dynamic migration of applications                            60
5.5 Enabling of emerging technologies on domain-specific architectures | 61

Chapter 6. High Performance processors using Stacked 3D Tunnel FETs

6.1 Introduction                                                     67
6.2 Motivation                                                       69
  6.2.1 Thermal constraints on processor execution                   70
  6.2.2 Yield constraints on processor design                       70
  6.2.3 Opportunities with TFET processors                         71
6.3 Design Space Analysis of High Performance Systems                72
  6.3.1 Yield aware stacking of processors                          73
  6.3.2 Modeling thermal distribution across multicores             75
  6.3.3 Variation in microarchitecture                             76
6.4 System Infrastructure                                             77
  6.4.1 Architectural simulation setup and benchmarks               77
  6.4.2 Modeling thermal variation                                 78
  6.4.3 Scheduling diverse workloads on a stacked CMOS-TFET multicore | 79
6.5 Results                                                         82
  6.5.1 Optimal operating points in the design space                83
## List of Tables

4.1 System Parameters ......................................................... 27  
4.2 Evaluated multicore configurations. The performance of these schemes are presented in Section 4.3.13. ......................................................... 39  
4.3 System parameters. .......................................................... 40  
6.1 Technology and System Parameters ......................................... 72  
6.2 Core and memory configurations ............................................. 77  
6.3 Technology parameters. ..................................................... 77  
6.4 Configuration of the evaluation platform. .................................... 83  
6.5 Best performing configuration for each workload .......................... 86  
7.1 System Parameters. ............................................................ 97  
7.2 Evaluated Cache configuration. Capacity is normalized to HP CMOS SRAM. ................................................................. 97  
7.3 Average energy, performance, EDP, and ED$^2$ results normalized to HP CMOS cache in each domain. ......................................................... 98  
7.4 The best cache types and the corresponding evaluation normalized metrics for (top) MiBench and (bottom) SPEC benchmarks. ......................................... 99
List of Figures

2.1 Switching time and energy of CMOS and several beyond-CMOS devices for a 32-bit full adder circuit in 15 nm node (From [80]). ........................................... 7
2.2 (a) Structure of an UTB 3D TFET, (b) structure of a 3D Si FinFET. .......... 8
3.1 Device-to-architecture abstraction model for Steep Slope Processors ........ 11
3.2 (a) 9-stage ring-oscillator used to model the critical path in the processor, (b) frequency vs $V_{cc}$ behavior for the Atom processor, and (c) frequency vs power behavior for the Atom processor. ......................................................... 12
3.3 Variation of CMOS and TFET core power with frequency for a simple (Atom-like) core and a complex (Ivybridge-like) core. ......................................................... 14
3.4 Impact of effective $V_{th}$ shift on transistor switching delay for a) TFETs and b) Si FinFETs. The effective $V_{th}$ shift in TFET is a reflection of the gate-contact work function variation. ......................................................... 15
3.5 Dependence of variation on supply voltage ........................................... 17
3.6 Dependence of variation on frequency .................................................. 17
3.7 Power-frequency curves for CMOS and TFET processors with and without the effects of variation. ................................................................. 18
3.8 Variation of total critical path delay ..................................................... 19
3.9 Variation of total core power ............................................................. 19
4.1 A homogeneous CMOS multicore (left) and a heterogeneous CMOS-TFET multicore (right) operating at dark/dim silicon settings. ....................... 21
4.2 Leakage energy savings as a result of migration to TFET cores under S1, S2, and S1+S2 strategies. ................................................................. 29
4.3 Dynamic energy savings as a result of migration to TFET cores under S1, S2, and S1+S2 strategies. ................................................................. 30
4.4 Relative time threads spend executing on TFET cores under S1, S2, and S1+S2 strategies. ................................................................. 31
4.5 EDP improvements as a result of migration to TFET cores under S1, S2, and S1+S2 strategies. ................................................................. 33
4.6 Variation in CMOS and TFET processor power at different frequencies for different number of cores. ................................................................. 35
4.7 Performance comparison of the best and worst thread mappings on our CMOS-TFET heterogeneous multicore. ................................................................. 43
4.8 Performance of our CMOS-TFET multicore with and without dynamic work and power partitioning schemes. ................................................................. 44
4.9 Power consumed by heterogeneous (top) and homogeneous (middle) systems, and number of active cores (bottom) over time ($wupwise$). ................. 45
4.10 Performance of CMOS-TFET multicore under different power budgets($wupwise$). 47
4.11 Performance of CMOS-TFET multicore for varying number of CMOS cores. . 48
4.12 Weighted speedups obtained with static and dynamic power partitioning methods 49
5.1 Variation of energy delay product (normalized to single issue CMOS) for CMOS and TFET cores with varying issue widths .................................................. 53
5.2 Permissible states in the frequency-issue-width design space at different thermal limits .............................................................. 55
5.3 Comparison of thermal profiles of cores for different thermal limits .......................................................... 56
5.4 Runtime Slack Estimation ................................................................. 58
5.5 Speedup obtained on the heterogeneous multicore due to static mapping w.r.t the best homogeneous CMOS configuration for thermal limits of 330K, 340K, 350K .................................................................................. 60
5.6 Normalized energy of the heterogeneous multicore due to static mapping w.r.t the best homogeneous CMOS configuration for thermal limits of 330K, 340K, 350K .................................................................................. 61
5.7 Speedup obtained on heterogeneous multicore due to DynMap .................................................................................. 62
5.8 Normalized energy in heterogeneous multicore .................................................................................. 62
5.9 Block diagram for computation of Euclidean distance .................................................................................. 63
5.10 Normalized delay, power, energy and EDP of TFET, iso-performance CMOS and iso-voltage CMOS accelerator designs. All results are normalized w.r.t the TFET design. .................................................................................. 64
5.11 Comparison of iso-performance, iso-voltage and peak-performance CMOS with TFET designs for a 32 point FFT accelerator .................................................................................. 65
6.1 a) and b) Demonstration of yield and thermal limits on performance scaling .................................................. 71
6.2 Number of core layers required to realize a range of functioning cores for different area footprints. .................................................................................. 74
6.3 a) and b) Delineation of design space attainable by CMOS(red), TFET(blue), both (green) and neither (black) cores to obtain peak performance, .................................................................................. 75
6.4 Variation in thermal hotspots on a layer of a system of 3D stacked cores with shared L3 cache. .................................................................................. 79
6.5 Characterization of SpecCPU2006, Parsec & Splash2 workloads based on scalability and memory utilization .................................................................................. 80
6.6 Different operating states of the heterogeneous multicore .................................................................................. 82
6.7 a) and b) Mean speedup of different applications in the splash2 and parsec suites respectively. .................................................................................. 84
6.8 a) and b) Mean power savings obtained in parsec and splash2 suites respectively .................................................................................. 84
6.9 Relative performances of 3D stacked CMOS and TFET configurations .................................................................................. 85
6.10 Variation of performance improvement of TFET core .................................................................................. 86
6.11 Comparison of performance speedup of CMOS and TFET cores for different microarchitectural configurations .................................................................................. 88
6.12 Performance comparison of homogeneous CMOS and TFET multicore with a heterogeneous 3D configuration with HAS, consisting of 1 CMOS layer and 7 TFET layers. .................................................................................. 89
7.1 Variation of EDP in omnetpp-lbm workload with varying cache area ratios. .................................................................................. 100
7.2 EDP of multiprogrammed workloads in low energy domain with best private and shared caches (normalized to iso-area shared HP cache). .................................................................................. 101
<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.3</td>
<td>L2 cache miss rates of HP, STT-Area, and TFET (SPEC on low energy domain).</td>
<td>102</td>
</tr>
<tr>
<td>7.4</td>
<td>Average memory access time of STT-Area and TFET normalized to HP (SPEC on low energy domain).</td>
<td>103</td>
</tr>
<tr>
<td>7.5</td>
<td>IPC of STT-Area and TFET normalized to HP (SPEC on low energy domain).</td>
<td>104</td>
</tr>
</tbody>
</table>
I have had the privilege of working with numerous people over the course of my PhD, both at Penn State as well as during my internships at IBM and Intel. In addition, I have had the opportunity to establish both personal and professional relationships with several others. I shall always be grateful for the discussions that we have had and for the assistance and the advice that they provided.

I wish to thank Emre Kultursay and Vinay Saripalli for their contributions to the material in chapters 3, 4, 5 and 7, Huichu Liu and Xueqing Li for their contributions to material in chapters 3 and 6 and also Moon Seok Kim and Nandhini Chandramoorthy for their contributions to Section 5.5 in Chapter 5. In addition, I would like to thank Ravindhiran Mukundrajan, Niranjan Soundararajan, Cong Xu, Raghav Pisolkar, Jagadish Kotra, Matthew Cotter and Siddharth Advani for their assistance in other projects that I worked on.

I had the opportunity to complete internships with highly regarded groups at IBM T.J Watson Research Center, IBM SRDC and Intel Components Research. I would like to acknowledge the role played by Pradip Bose, Alper Buyuktosunoglu, Phil Emma, Robert Montoye, Toshiaki Kirihata, Jack Golz and Srivatsan Chellappa at IBM and Ian Young, Uygar Avci and Daniel Morris at Intel.

I am deeply grateful to my advisor Prof. Vijaykrishnan Narayanan for his guidance, encouragement and unstinting support through the course of my Ph.D. I would also like to express my gratitude to the other members of my thesis committee, professors Mahmut Kandemir, Anand Sivasubramaniam, Jack Sampson and Jeya Chandra for their advice and assistance in
times of need. I would also like to thank professors Suman Datta and Yuan Xie for their valuable inputs.

Finally, I would like to thank my parents and family members who have remained steadfast in their support during the pursuit of my doctoral degree.
"Technology is a word that describes something that doesn't work yet"

-Douglas Adams
Chapter 1

Introduction

Although advances in transistor scaling has enabled more and more devices to be accommodated on a single chip, limited power budgets have prevented these advances from being translated into absolute performance. With technology such as the 72 core Tile72 chip and the Gx100 with 100 cores on chip, there is a new problem that designers are now forced to address— that of increased power density [73]. Increased power density can also result in an undesirable increase in the chip temperature, resulting in thermal emergencies. This in turn can severely affect the reliable functioning and lifetime of the chip. In order to mitigate this problem some of the cores on the chip have to be turned off or run at lower operating voltages. While there have been works such as [45] that try to exploit the fact that only a subset of cores can run at any time, in most cases this results in a sub-optimal utilization of the available hardware. Consequently, efforts are being made to achieve higher speedups at supply voltages ($V_{DDs}$) that continue to drop with every generation. Thus, as chip designers attempt to move tasks with large computational demands into increasingly power-sensitive and energy-sensitive domains, the relationship between supply voltages, performance, and power at lower operating voltages assumes greater significance. While the pace of device innovations have enabled designers to keep up with Moore’s law until now, this lowering of $V_{DD}$ brings about a new set of problems due to the inherent physical characteristics of MOS transistors. Innovations in CMOS transistor
technology such as trigate FinFETs have bucked this trend to an extent; however they too face the same fundamental limitations as previous technologies.

An effective representation of the tradeoffs between power and performance at voltages that approach or dip below the transistor threshold voltage ($V_{th}$) is provided by a device’s sub-threshold slope. It is known that the sub-threshold slope of even the most state-of-the-art transistors is limited to 60 mV/decade, on account of physical limitations in the MOS charge transport mechanism. This causes inefficient switching characteristics as supply voltages are lowered below 0.5V. This has an adverse impact on performance, due to the slow switching speed, and on the power, due to the increased off-state leakage of the transistor. Consequently, there is a need to explore new device technologies that do not suffer from this shortcoming, in order to sustain the performance improvements demanded from each subsequent processor generation.

Various emerging technologies have been touted as viable replacements for CMOS, and substantial improvements in performance, power and reliability have been demonstrated in some cases. However, simply designing a transistor that outperforms CMOS under certain conditions does not necessarily result in a faster or a more power-efficient processor. For instance, reducing the device power cannot come at the cost of severely hindering the switching speed and compromising the peak performance of the processor. In this context, a device-to-architecture abstraction model is essential to analyze how these device characteristics translate to the system and architecture-level. This would enable us to simultaneously analyze innovations in both the device and architecture by means of common metrics such as performance and energy. In addition, it is also essential to carry out a comprehensive study of the various applications that are slated to run on the proposed system and explore various application mapping techniques that fully exploit the advantages provided by adoption of the new technology. Such a study requires
detailed analyses at multiple levels of abstraction, ranging from the device up to the system and application level, in order to yield tangible benefits for the end user. In effect, the aim of this dissertation is to attempt to answer the question: 'Does a better device make a better computer?'

This dissertation thus addresses the following topics:

- An exploration of the various device technologies that could be possible complements, or replacements for CMOS technology. In particular, we show how the Heterojunction Tunnel Field Effect Transistor (HTFET or TFET) is a promising candidate for this purpose.

- A device-to-architecture abstraction model that enables us to extrapolate various processor architectures, given the characteristics of a single constituent device.

- An examination of multicore systems designed using both CMOS and TFET processors. We examine the feasibility of performance constrained power optimization, as well as maximizing the performance under fixed power budgets.

- A study of the interactions between device and micro-architectural heterogeneity, in terms of issue width, pipeline depth and other factors, as well as different static and dynamic schemes to optimally map applications onto the best possible core configuration.

- Design of domain-specific customized accelerators for computer-vision-based applications, using TFET technology with superior energy-efficient characteristics in comparison to CMOS accelerators.

- Exploring the possibility of using device-heterogeneous multicores to design 3D stacked CMPs by mitigating thermal and yield constraints faced by conventional device-based architectures.
• Examining the various aspects involved in the design of on-chip caches, both from a technology as well as a policy standpoint.

The chapters are organized as follows. Chapter 2 provides a brief overview of the various emerging technologies and their potential applications in processor architecture. Chapter 3 explains in detail the device-to-processor abstraction model used in this work. It also looks at incorporating device-level factors such as noise and variability that have an impact on the entire system in the form of process variation, for example. Chapter 4 proposes a device-heterogeneous multicore and studies the techniques used to best utilize this multicore under performance and power constraints for a variety of workloads comprising of single and multithreaded applications, as well as single and multi-programmed workloads. Chapter 5 looks at combining device and architecture heterogeneity. It proposes static and dynamic schemes, where runtime application characteristics can be used to determine not only the optimal device technology, but also the optimal architecture configuration that yields the most energy-efficient performance. It also examines the use of emerging technologies in the design of customized accelerators for highly energy efficient realization of hardware for computer vision and image processing applications. In Chapter 6, the adoption of state-of-the-art architecture techniques such as 3D stacking technologies is studied, in conjunction with CMOS-TFET device heterogeneity. Chapter 7, extends the study of emerging technologies in architectures to the memory hierarchy. In particular, it comprises of a design-space evaluation of last-level caches for different application domains, jointly considering the adoption of new technologies as well as architectural techniques. Chapter 8 examines the prior work in various areas touched upon in this dissertation, ranging from
emerging technologies to application scheduling on heterogeneous architectures. It also highlights the distinguishing aspects of the prior work from the work in this dissertation. Finally, chapter 9 concludes with a summary of all the work described in this dissertation.
Chapter 2

The landscape of emerging technologies - An overview

There have been several efforts at carrying out a characterization of emerging device technologies at a device and circuit-level. In 2013, Nikonov et. al [80] benchmarked a range of these devices by comparing the energy-performance characteristics of a 32-bit adder built using these technologies, in comparison to low power (LP) and high performance (HP) CMOS technology, as shown in Figure 2.1. These devices include Homojunction and Heterojunction Tunnel FETs, Graphene Nano-Ribbon FET and Bilayer PseudoSpin FETs.

Among these devices, it can be observed that there are several potential candidates that are either faster than Low-Power (LP) CMOS or more energy efficient than High Performance (HP) CMOS. These devices, fall under the category of a special class of devices called Steep Slope Devices. In this section, I shall provide a brief overview of the properties of these devices as well as their benefits and limitations.

2.1 Basic background on Steep Slope devices

Steep slope devices have been proposed as an alternative to counter the 60 mv/decade subthreshold limitation that restricts the scaling of conventional CMOS transistors. This property of CMOS causes an exponential increase in leakage current as supply voltage ($V_{dd}$) scales to near- and sub-threshold values and threshold voltage ($V_T$) remains roughly constant. In contrast, steep-slope devices do not suffer from this sub-threshold limitation on account of their charge...
transport mechanism which involves tunneling through the intrinsic region. Hence, at near-threshold and sub-threshold voltages, these steep slope devices have the potential to outperform CMOS devices to several orders of magnitude. The Tunnel Field Effect Transistor or TFET, seen as one of the most promising post-CMOS alternatives, belongs to this class of Steep Slope Transistors.

There have been numerous experimental demonstrations of these tunneling (TFET) devices. The first such demonstration was an In$_{0.53}$Ga$_{0.47}$As homojunction TFET (HomjTFET) [77] which illustrated the concept of a vertical interband tunneling transistor. The problem of low $I_{on}$ in homojunction TFETs was overcome by using a GaAs$_{0.1}$Sb$_{0.9}$/InAs heterojunction TFET (HTFET). On account of the P-N heterojunction being staggered, and InAs having a lower bandgap ($E_G$), this device was observed to have a higher $I_{on}$ than a homojunction TFET. A vertically-oriented, gate-all-around silicon nanowire was demonstrated, showing 50 mV/decade sub-threshold slope over 3 decades of drain current in [32]. Further, a process flow for the
creation of a side-gated vertical-mesa TFET which can be scaled down to achieve an Ultra-Thin-Body (UTB) double-gated structure has also been demonstrated [75, 76].

Figure 2.2(a) shows the 3D schematic of a vertically-oriented Ultra-Thin Body (UTB) HTFET, the fabrication of which has been demonstrated in [76]. Modeling of such a transistor has been performed for different material systems using advanced atomistic simulations [68, 5]. This TFET device is compared against bulk-CMOS devices in 20nm CMOS technology in [90]. However, beyond the 20nm node, designers in industry have adopted the tri-gate transistor technology (i.e., FinFETs) for realizing CMOS technology [20]. The 3D schematic of a silicon FinFET is shown in Figure 2.2(b). Hence, all subsequent comparisons with CMOS technology in this dissertation have been made with the state-of-the-art trigate FinFET designs.

As can be observed in Figure 2.1, Heterojunction Tunnel FETs (TFETs) are one of the most promising steep slope devices, both in terms of subthreshold operation and high speed switching operations and they have been shown to scale well into future process nodes [65].
2.1.1 The Manufacturing Barrier: Integrating TFET and FinFET Technologies

From an integration perspective, TFET and FinFET devices can be fabricated on a single chip in a monolithic fashion using a silicon substrate. Such integration has already been demonstrated through experimental fabrication of InGaAs-based Quantum-Well FETs (QWFETs) on silicon substrate using a metamorphic buffer layer growth scheme [23]. There have also been experimental demonstrations for III-V n-FET and SiGe p-FET co-integration in [22] and for vertical III-V TFET integration on Si substrate in [87] and [102], which makes heterogeneous integration of TFET and CMOS devices possible on the same die.

TFETs do however, suffer in some aspects when compared to CMOS technology. As the supply voltage is increased, the inherent limitation in the TFET charge-carrying mechanism causes the current to saturate above a certain operating voltage. Due to the saturation of the tunneling current, the switching delay remains constant beyond a certain supply voltage. At a processor level, this translates to a limited range of operating frequencies for TFET transistors.

It is possible to tune TFET device characteristics to an extent to improve frequency and power responses through altering channel length [65]. By using a low static power (LSTP) TFET with an increased channel length, TCAD [1] simulations show that it is possible to realize the drive current required while drastically reducing overall power consumption by 2× over existing device models [54].
Chapter 3

Design of Heterogeneous CMOS-TFET multicores
- From device to processor

3.1 Introduction

Although several of the emerging devices described in Chapter 2 are considered promising candidates for future processor design, the existing CMOS technology still continues to have several characteristics that enable it to reach performance points that cannot be attained by these new devices. Consequently, these devices cannot be used as mere *drop-in replacements* for CMOS technology. Hence, on account of their differing points of optimal operation, it may be necessary to rethink the design at a circuit and architecture-level in order to incorporate these devices effectively into processor design. To this end, an abstraction model is proposed, where it is possible to characterize the behavior of entire processors from that of a single transistor.

3.2 Device-to-processor abstraction model

In order to implement circuit and architecture designs using emerging device technologies, a device-circuit-architecture modeling framework has been proposed to bridge the different design layers, as shown in Fig. 3.1. A lookup-table based TFET Verilog-A model has been used to accurately model the device performance in TFET prototype circuit demonstration and performance benchmarking. Here, the device characteristics across a wide range of operation
voltages can be directly obtained from full-band atomistic simulations [6] or from calibrated TCAD Sentaurus simulations [59, 89].

Based on the device models, various circuit designs using steep slope TFETs have been explored for ultra-low power digital and analog/RF applications [89, 82, 103, 64] by taking advantage of the steep sub-threshold slope-induced energy-efficiency benefits that these devices provide. Studies of TFET electrical noise modeling [83] and variation impacts [6, 89] provide more insights for design tradeoffs between energy efficiency and reliability.

Fig. 3.1 Device-to-architecture abstraction model for Steep Slope Processors
An architecture-level abstraction from the aforementioned circuit design can be obtained in different ways, depending on the nature of the processor configuration. As a first-order analysis, the critical path delay of a CMOS-based in-order core can be obtained by calibrating the delay of a series of cascaded FO4 inverters to form a ring oscillator with the frequency of an existing Intel Atom processor configuration. The total core power is obtained by multiplying the individual transistor power by the total number of transistors in the processor. The TFET processor critical path delay and power at different supply voltages are obtained by scaling the corresponding FinFET core parameters at that particular voltage.

![Diagram of a 9 Stage Ring Oscillator](image)

**Fig. 3.2** (a) 9-stage ring-oscillator used to model the critical path in the processor, (b) frequency vs $V_{cc}$ behavior for the Atom processor, and (c) frequency vs power behavior for the Atom processor.

The maximum operating frequency of a processor is set by its critical path, which is modeled using a ring-oscillator chain of 9 NAND gates each driving 5 NAND gates, as shown in Figure 3.2(a), resulting in a 45 FO4 delay for the ring-oscillator. Using a 45nm bulk-CMOS model [112], this logic chain is able to accurately model the $V_{cc}$-frequency behavior of the Intel
Atom processor [19], as shown in Figure 3.2(b). The switching energy per transistor is estimated from the ring oscillator. Typical switching factors obtained during logic benchmarking range from 12.5% (in logic intensive designs) to 50% (in arithmetic intensive designs), making an average activity factor of 30% a reasonable assumption [109]. Using the Atom processor’s transistor count of 47 million along with this transistor switching activity factor, its power-frequency characteristics can be modeled to a reasonable degree of accuracy.

Assuming an average IPC of 1, which can be representative of a standard workload run on a 2-issue Intel Atom processor, and by extrapolating from the 45 nm technology node to the 20 nm node, the energy per instruction of the target processor is obtained using both Silicon FinFET and heterojunction TFET technology at 20nm using techniques similar to those described in [95].

On the other hand, due to the diversity in critical paths in out-of-order cores, and the corresponding impact of non-logic components, especially that of interconnects, a simple ring-oscillator model may not be sufficiently accurate. In addition, wire delay and power which forms an important constituent of the overall processor delay and power, cannot be accurately estimated using the ring-oscillator model. Hence, power and timing evaluations of existing processor designs such as Intel IvyBridge are carried out using an architectural simulator, namely Sniper-6.0 [13] which is integrated with McPAT-1.0 [61], a power and timing estimation tool. The McPAT source code is modified to output the critical path delay and power for a 20nm FinFET processor design and obtain the corresponding TFET core numbers by scaling the device parameters as in the previous case. The power and delay of the most significant microarchitectural components is obtained by instrumenting McPAT and add a model for wire power and delay for both FinFET and TFET technologies.
Figure 3.3 shows the power-frequency tradeoffs for a 2-issue Atom-like core and a 4-issue Ivybridge-like core, when both are realized using CMOS and TFET technology efficiency. The crossover frequency ($f_c$) point is lower for the Atom core compared to the Ivybridge core. This is because wire delay and power become more of a factor in the more complex configuration.

![Figure 3.3 Variation of CMOS and TFET core power with frequency for a simple (Atom-like) core and a complex (Ivybridge-like) core.](image)

While the system-level behavior of steep-slope devices such as Tunnel FETs are shown to be promising based on the models described above, it is also necessary to consider non-idealities in their characteristics, in order to carry out a fair comparison with existing technologies. Consequently, the following sections describe the impact of physical factors such as variation and technology scaling of devices, and their translation to the architecture level of abstraction.
3.3 Variation in steep-slope multicores

Steep-slope devices are prone to process variations due to their operation at reduced supply voltage and the steep transition of their current characteristics. For III-V HTFET, variation sources that can alter the tunneling barrier width can cause a significant $I_{on}$ fluctuation given the exponential current dependence on the tunneling barrier width [89, 6]. Variation sources such as fluctuations in source doping, oxide thickness, gate-contact work function, left/right gate edge overlap, body thickness have been investigated from the device to circuit level [89], among which the work function variation dominates, considering the reported process data [6]. Fig. 3.4 shows the transistor delay fluctuation corresponding to the threshold voltage shift caused by the gate work function variation, based on the Verilog-A models of $L_g=20$ nm HTFET and Si FinFET. HTFET exhibits an overall lower delay variation compared to Si FinFET for $V_{dd} < 0.5V$ while Si FinFET operates in the near- or sub-threshold region.

Fig. 3.4 Impact of effective $V_{th}$ shift on transistor switching delay for a) TFETs and b) Si FinFETs. The effective $V_{th}$ shift in TFET is a reflection of the gate-contact work function variation.
3.3.1 Observing device variations at the architecture level

Based on the different sources of variations at the device level, the variations are classified as global (constant) and random variations. As observed, the variation in work function, an inherent atomistic property of transistors, dominates the overall variation for TFETs [6]. In a similar manner, the corresponding variation in FinFETs can be modeled as a zero mean Gaussian with a work function variation of around 50 mV, as demonstrated in [93]. The remaining sources of variation, allied with variation components localized to a section of the wafer, can be assumed to encompass 3-5% of the total variation [52].

In order to model the critical path delay of a variation-affected processor, a number of such variation-affected transistors are cascaded together. The exact length of this critical path is determined by a critical path analysis using Fabscalar [16]. This tool generates synthesizable HDL code for different micro-architectural configurations, by varying parameters such as the issue width, pipeline depth and number of execution units.

Figures 3.5 and 3.6 show the dependence on the worst case (3σ) variation in delay on supply voltage and frequency respectively, for both FinFET and TFET-based processors. The processor model used was a 4-issue Ivybridge configuration. These figures lead us to draw the following conclusions. Firstly, TFET processors are less susceptible to variations at low V_{dd} than their FinFET counterparts. Also, the variation behavior in the region of optimal operation of both core types are roughly similar (within ±10% of each other).

3.3.2 Variation mitigation using voltage scaling

In order for a system to meet its specified performance criterion in the presence of variation, it is possible to boost the supply voltage to compensate for the additional worst-case
variation-induced delay. Doing so, however, comes at the cost of additional dynamic and leakage power. As shown in Figure 3.5, both the distribution of variation and its worst case ($\mu \pm 3\sigma$), depend on the target supply voltage. Further, in case of TFET processors, this can also shift the effective crossover point, i.e. $f_c$, below which TFET cores are more power efficient than CMOS cores. Since the variation of TFETs becomes higher than CMOS for supply voltages above $V_t$, the crossover point, which falls in this super-threshold region is shifted to the left due to the additional voltage boost required by TFETs to compensate for the increase in worst-case delay. Figure 3.7 shows the effect of these overall variations on the TFET-CMOS crossover plot.

### 3.4 Impact of technology scaling

The base device models are based on 20 nm technology node simulations calibrated with fabricated devices [91]. In addition, TFET device scaling is used to model technologies up to the 10 nm node [65].

Figure 3.8 shows the scaling of the critical path delay when extrapolated to future technology nodes. Comparisons are made between the ITRS 2012 [43] roadmap projections for Si
Fig. 3.7 Power-frequency curves for CMOS and TFET processors with and without the effects of variation.

FinFET and simulation results for HTFET. The HTFET device models for 14 nm and 10 nm technology nodes are generated from simulations using the TCAD Sentaurus device modeling tool [1]. The supply voltage $V_{cc}$ corresponding to each technology node is $V_{cc} = 0.72 \, V$ (22 nm node), $0.67 \, V$ (14 nm node), $0.55 \, V$ (10 nm node) for Si FinFET technology; and $V_{cc} = 0.4 \, V$ (22 nm node), $0.35 \, V$ (14 nm node) and $0.3 \, V$ (10 nm node) for HTFET technology. In a similar manner, Figure 3.9 shows the scaling of the total core power for each of the above technology nodes.

One of the major limitations with TFET processors at the current (20 nm) technology node is their relatively low peak performance. The saturating nature of TFET tunneling current forces the peak frequency to be restricted to around 1.7-1.8 GHz, which is far below what CMOS processors are capable of attaining. However the minimum switching delay of the device reduces with subsequent generations, enabling future TFET processors to operate at much
higher frequency. Although there is a proportional decrease in FinFET switching as well, the non-scaling of wire-delays causes the frequency gap between CMOS and TFET processors to shrink with every generation. This is because the gap in the critical path delay between CMOS and TFET goes on decreasing with technology and by the 10 nm node, TFET cores can attain 95% of the peak performance of CMOS cores, as compared to 60% for the current (20 nm) technology node. In addition, TFETs are clearly becoming more and more power efficient w.r.t CMOS with each subsequent generation. Thus the range of applications where TFETs can act as a viable power-efficient CMOS replacement goes on expanding as transistors continue to scale.
Chapter 4

Architectural exploration of device-heterogeneous multicores

This chapter examines the potential of using steep slope device-based architectures to overcome problems faced by traditional CMOS technology, and to achieve unprecedented improvements in performance and energy efficiency.

While each technology generation enables us to pack more cores on the same die, thermal and power delivery constraints have precluded any scaling in the power budget available to these cores. This forces cores to operate at very low voltages (termed as “dim silicon”) to stay within the allotted power budgets. Unfortunately, low-voltage operation of silicon CMOS technology is extremely energy-inefficient. As the supply voltage approaches the threshold voltage, the transistor delay increases rapidly, resulting in a drop in the clock frequency. To avoid inefficient low-voltage operating points, CMOS multicores typically power on only a subset of the available cores and turn off the remaining cores (termed as “dark silicon”). On the other hand, the high degree of efficiency exhibited by TFET cores at low voltage points enable us to tackle several problems in architecture, where it is possible to trade off peak single-threaded performance for parallelism, thus resulting in huge performance and energy improvements.
4.1 Comparison of CMOS/TFET cores from the perspectives of dark and dim silicon

Figure 4.1 compares a homogeneous CMOS multicore with a heterogeneous CMOS-TFET multicore in both dark and dim silicon configurations. In a dark silicon setting, the heterogeneous multicore can match the performance of the homogeneous multicore as long as it contains enough CMOS cores (1 vs. 4). In a dim silicon setting, the heterogeneous multicore can outperform the homogeneous multicore by either using the same number of TFET cores at a higher frequency (2 vs. 5) or more TFET cores at the same frequency (2 vs. 6). Dimming the CMOS multicore can enable more cores to be turned on, but forces these cores to operate at extremely low frequencies (3). The heterogeneous multicore can match the dark silicon performance of the homogeneous configuration because it can activate the same number of CMOS
cores at high frequencies. In addition, it can outperform the homogeneous processor in a dim silicon setting because it employs low voltage optimized TFET cores. The heterogeneous multicore can thus use the same power budget to either turn on more cores at the same frequency or use the same number of cores at higher frequencies. Although a CMOS-TFET heterogeneous multicore can operate efficiently on both dark and dim silicon configurations, an application could prefer one configuration over another owing to factors such as peak instruction throughput and thread and core scalability. Therefore, mapping applications on a heterogeneous system poses several interesting questions. Given a number of applications to execute, how many cores of each type and how much power should each application be allocated? How should the applications’ threads share these resources? In order to answer these questions, it is possible to formulate an optimization problem by reducing power consumption under performance constraints, and improving performance under power constraints. In this chapter, I examine this problem from these two perspectives and propose various static and dynamic application scheduling and mapping schemes. I also demonstrate our schemes’ effectiveness in improving the energy efficiency of heterogeneous CMOS-TFET multicores to well above the corresponding homogeneous configuration.

4.2 Energy optimizations on CMOS-TFET multicores

4.2.1 Voltage scaling for Multithreaded Applications

Applications do not show the same behavior throughout their execution, typically demonstrating a dynamic fluctuation in instructions per cycle (IPC). A low IPC typically means that the application is in a memory-bound phase and spends relatively more cycles waiting for a response
from the memory hierarchy. In this scenario, dynamically reducing the voltage/frequency of the core executing the application can result in a significant reduction in its leakage and dynamic energy consumption with a relatively small degradation in performance. Therefore, operating cores always at the maximum voltage/frequency is not necessary and an Energy-Delay Product-aware (EDP-aware) Dynamic Voltage and Frequency Scaling (DVFS [70]) can achieve higher energy efficiency. Another optimization that is shown to provide energy savings is barrier-aware DVFS [62], which addresses a problem that is specific to multithreaded applications. Multithreaded applications typically use barriers for synchronization, where a thread that arrives at a barrier must wait until all other threads also reach the barrier. Between any two barriers, the performance of the application is limited by the performance of the slowest thread in that region, resulting in all other threads waiting idle at the barrier. Barrier wait times of these threads can be significant if the workload is not evenly distributed across threads. Observing the idle wait times at barriers due to an unequal workload distribution, barrier-aware DVFS can reduce the voltage/frequency of each core individually such that it reaches the barrier at around the same time as the slowest thread. This reduction in supply voltage directly translates into lower leakage and dynamic energy without any significant performance degradation, thereby improving energy efficiency.

4.2.2 EDP-Aware DVFS

DVFS or Dynamic Voltage/Frequency Scaling is a technique that exploits the dynamic changes in the behavior of applications throughout their course of execution, by dynamically tuning the supply voltage and operating frequency of the core. When an application enters a memory-bound phase, its performance is mostly limited by the memory access latency, and
therefore, the frequency at which the corresponding core is running has little impact on performance. In this case, voltage/frequency of the core can be reduced in order to reduce leakage power and dynamic energy. On the other hand, when the application enters a compute-bound phase, performance is mostly determined by the frequency of the core, so higher frequencies are desired.

DVFS [70] can be applied to exploit this dynamic variation, by optimizing several metrics such as IPC or EDP (Energy-Delay Product). In order to improve the energy efficiency of multicore processors, an EDP-aware DVFS scheme is used, which is a modified version of the algorithm presented in [96]. This algorithm dynamically monitors EDP and performs DVFS actions at the end of every epoch. At the \(i\)th epoch, a core can be in any one of \(N_L\) DVFS levels, which means that during that epoch, the core executes at voltage \((V_i)\) and frequency \((f_i)\) determined by that DVFS level. At the end of each epoch, one of the following DVFS decisions is carried out: (i) the core moves one DVFS level up (increased frequency and voltage) (ii) the core moves one DVFS level down (decreased frequency and voltage) or (iii) the level remains the same. This DVFS decision is performed based on the EDP value observed during the last two epochs and the latest DVFS action performed. At the end of epoch \(i\), the EDP of the core during that epoch \((EDP_i)\) is calculated and compared against the EDP at the previous epoch \((EDP_{i-1})\). If an improvement in EDP is observed to be above a certain threshold, then the DVFS decision performed in the previous epoch is carried out again for this epoch. However, if the EDP is observed to degrade beyond a similar threshold, then a DVFS decision is made in the direction opposite to the one taken in the previous epoch. It should be noted that, this threshold is set such that the algorithm still preserves its ability to accurately track the dynamic change in behavior of applications.
4.2.3 Barrier-Aware DVFS

Barriers are the most widely used synchronization primitives in multithreaded applications. A barrier is essentially a mechanism to prevent the progress of threads beyond the barrier until all threads reach the barrier. A thread that reaches the barrier early must wait for all other threads to arrive at the barrier. Barriers are typically implemented as shared counters that are incremented in a critical region whenever a thread arrives at the barrier. All threads waiting at a barrier continuously check the value of this counter and continue only when it becomes equal to the number of threads that must synchronize at the barrier. This requires threads to continuously read their local copies of the shared counter, doing no useful work and consuming leakage and dynamic energy.

Energy overheads associated with barrier synchronization are proportional to the barrier wait times of threads. As observed in [9], an imbalance in workload distribution across threads can result in large barrier wait times. A solution to this problem is to use barrier-aware DVFS. This ensures that, at each barrier, the voltages and frequencies of cores are dynamically scaled such that faster threads do not arrive at the barrier early, but instead, they arrive at the barrier at around the same time as the slowest thread. Using this technique, the wait times of threads at the barrier are minimized, and as a result, redundant leakage and dynamic energy consumption spent at the barrier is avoided. In order to obtain energy savings using barrier-aware DVFS, the discrepancy in thread execution times between two barriers, and in turn the expected barrier stall times when all cores run at the maximum frequency, must be accurately predicted. Accurate barrier stall time predictions can be performed by applying any of the techniques in [9, 12, 60, 62]. The objective of this work is not to evaluate the accuracy of these techniques, but to show
the potential benefits that can be obtained by using TFET cores under any barrier-aware DVFS technique, an oracle predictor that can always find the best voltage/frequency is preferred for each core so that the barrier wait times are minimized.

4.2.4 Thread Migration

Each of the DVFS schemes described above results in a scaling coefficient, $k_{edp}$ and $k_{barrier}$ respectively. The target frequency for each core is calculated by multiplying the maximum frequency with both these coefficients. As a result of applying EDP-aware DVFS and barrier-aware DVFS, during its course of execution, a thread runs at continuously changing voltage/frequency levels. As seen in Chapter 3, there exists a crossover frequency ($f_c$) below which TFET cores are more energy efficient than CMOS cores. Operation below this frequency is more energy efficient when run on TFET cores. Let the corresponding DVFS level be denoted as $L_c$. Thus, an energy-aware thread-to-core mapping mechanism should prefer to run threads on TFET cores if the corresponding DVFS level is less than $f_c$ and on CMOS cores if it is greater than $f_c$. In order to achieve this, a thread migration scheme is adopted. This scheme uses the DVFS levels of cores to move threads across CMOS and TFET cores. Our scheme migrates threads running on CMOS cores at DVFS levels lower than $L_c$ to TFET cores, and similarly, threads running on TFET cores at DVFS levels higher than $L_c$ to CMOS cores. Although, in theory, threads can be migrated across cores arbitrarily frequently, the cost associated with thread migration should be taken into account in practice. Therefore, in order to avoid excessive performance degradation due to frequent migrations, a guard band is applied at the crossover frequency.
4.2.5 Simulation Framework and Infrastructure

We carried out our experiments using the Simics [71] full system simulator. Our target system in this work contains an 8 core processor that consists of 4 CMOS cores and 4 TFET cores, running the Linux operating system. To fairly evaluate the benefits of using TFET cores over CMOS cores, a maximum of only 4 cores are in operation at any point of time and the rest of the cores are assumed to be in power-down mode. Most modern embedded processors are equipped with DVFS capability to allow the processor to operate in extremely low power modes as well as high performance modes. For example, the Intel XScale embedded processor implemented in 180nm technology provides 50mV DVFS stepping with an operating voltage range of 0.7V-1.8V, resulting in an operating frequency range from 200 MHz to 1000 MHz [18]. Targeting an embedded processor implemented in 22nm technology node, our supply voltages are much lower. Table 4.3 shows the details of the hardware configurations.

The schemes were evaluated using benchmarks from the SPLASH-2 [107] suite. Some of these workloads make extensive use of barriers for synchronization, which enables us to distinguish the benefits that arise from EDP-aware DVFS and barrier-aware DVFS.

Table 4.1 System Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Cores</td>
<td>4 CMOS + 4 TFET</td>
</tr>
<tr>
<td>L1 D/I-Cache</td>
<td>Private, 32KB each, 4-way set associative</td>
</tr>
<tr>
<td>L1 Access Latency</td>
<td>1 cycle</td>
</tr>
<tr>
<td>L2 Cache</td>
<td>Shared, 1MB, 16-way set associative</td>
</tr>
<tr>
<td>L2 Access Latency</td>
<td>10 cycles (at 1500 MHz)</td>
</tr>
<tr>
<td>Memory Access Latency</td>
<td>120 cycles (at 1500 MHz)</td>
</tr>
<tr>
<td>Epoch Size</td>
<td>10ms</td>
</tr>
</tbody>
</table>
In this work, per-core DVFS is achieved using on-chip voltage regulators. Moving from off-chip voltage regulators to per-core on-chip voltage regulators can provide DVFS transition times in the order of tens of nanoseconds [48]. This switching latency is much smaller than the epoch size used in our experiments.

Leakage energy of a core at each epoch depends on the leakage power at the corresponding frequency and the epoch duration. Core dynamic energy calculations are based on the dynamic energy per instruction values obtained from simulations and the number of instructions executed at each epoch. As will be shown in Section 4.2.6, the overhead due to thread migration is estimated to be around 20K cycles at 1500MHz. In order to ensure that this overhead remains a negligible fraction (< 2%) of the period of our thread migration decisions, the epoch size is fixed as 10 ms.

4.2.6 Experimental Results

The baseline system is assumed to have two DVFS mechanisms, namely, the EDP-aware DVFS (S1) and the barrier-aware DVFS (S2) as discussed in Sections 4.2.2 and 4.2.3, that can be applied to improve energy efficiency. The first scheme (S1), monitors the EDP of the application continuously and performs DVFS decisions accordingly, making use of the trade-off between performance and energy efficiency. The second scheme moves faster threads into lower DVFS levels to eliminate idle wait times at barriers. We also experimented with a combined scheme that employs both of these mechanisms (S1+S2). The results presented in this subsection are normalized with respect to the energy consumption and performance of our baseline system which executes threads only on CMOS cores (i.e., it does not utilize TFET cores), but still employs the same DVFS techniques to improve energy efficiency. Before experimenting with TFET cores,
in order to verify the correctness of the DVFS implementations, the baseline system with DVFS was compared against a system with no DVFS where all cores run at the maximum frequency and obtained an average EDP improvement of 11%. All reported results in this subsection are improvements on top of this 11%.

4.2.7 Energy Improvements

Figs. 4.2 and 4.3 respectively show the leakage and dynamic energy savings under S1, S2, and S1+S2 strategies. Using S1+S2, an average leakage energy improvement of 30% is obtained due to the very low leakage energy of TFET cores at low voltages. Dynamic energy is also improved by 17% on average, resulting in average total energy saving of 19%. The improvement in dynamic energy is associated with the lower supply voltage requirement of the TFET devices. In addition to this reduction in supply voltage, the lower off-currents of TFETs also result in significant additional leakage energy savings.

![Fig. 4.2 Leakage energy savings as a result of migration to TFET cores under S1, S2, and S1+S2 strategies.](image-url)
One observation from our energy results is that, there is wide variation across benchmarks in the energy savings obtained from considering each DVFS strategy individually. Considering S1, large benefits are obtained with low IPC applications such as lu, whereas for applications with already high IPC, such as radix, the gains are much lower. Similarly, not all benchmarks show high energy savings due to S2. For instance, in water-spa, up to 40% energy savings are obtained when barrier-aware DVFS is applied. However, in radix, this technique brings less than 1% energy savings. Actually, this behavior is expected as water-spa has a large imbalance in its workload distribution, whereas the uniform distribution of work across threads of radix results in almost no energy savings. It should be noted that, the overheads associated with barrier synchronization increase with the number of processors that meet at the barrier [62]. As a result, increasing the number of cores enhance the importance of barrier-aware DVFS, which can result in more cores running at lower frequencies. This is expected to result in the energy gains from migrating threads to TFET cores to become more significant in the many-core processors of the future when barrier-aware DVFS is employed.
It can be observed that the benefits of using TFET cores under S1+S2 exceeds the sum of the individual benefits of the two techniques. This is due to the constructive interference between the two DVFS techniques employed when energy improvement due to TFET cores is considered. As a result of applying the two DVFS techniques simultaneously, the time spent by threads at lower frequencies increases, thereby improving the leakage and dynamic energy savings from our thread migration scheme. The fraction of the total time spent by the threads on TFET cores for each benchmark is shown in Fig. 4.4. It can be observed that, as a result of applying our scheme, on average, 50% of execution time is spent running on TFET cores under S1+S2.

Fig. 4.4 Relative time threads spend executing on TFET cores under S1, S2, and S1+S2 strategies.

### 4.2.8 Performance Degradation

The energy savings that can be obtained from migrating threads across CMOS and TFET cores is given in the previous subsection. However, there are also overheads associated with thread migration across cores that must be quantified.
The worst case overhead is determined to be 2% of the granularity of each thread migration decision (i.e., the epoch size). The frequency of migrations was observed to be less than 1%, with a worst case migration frequency of 3% in \textit{fft}, which indicates that the proposed thread migration scheme does not suffer from frequent migrations across cores. Overall, we observed the performance degradations due to thread migration to be less than 1%.

4.2.9 Overall Impact on EDP

The combined effect of the leakage/dynamic energy improvements are analyzed and performance degradation results given in the previous two sections. The impact of thread migration on energy-delay product (EDP) for each benchmark is given Fig. 4.5 and shows that the EDP improvements follow the same improvement trend we have for energy. Maximum improvements are obtained with \textit{lu} and \textit{water-spa} benchmarks, which improve by 44% and 32%, respectively. However, the reasons for EDP improvement for these two benchmarks are different. In \textit{lu}, the benefits are mostly due to EDP-aware DVFS, whereas in \textit{water-spa}, most of the improvement is as a result of barrier-aware DVFS.

4.3 Performance Enhancement under Power Constraints for Heterogeneous CMOS-TFET Multicores

4.3.1 Motivation

In this section, an \textit{automatic runtime scheme} is proposed to extract high performance from heterogeneous CMOS-TFET multicores that operate under fixed power budgets. Our
scheme can be implemented as a part of the operating system (OS): (i) heterogeneous thread-to-core mapping (i.e., it can utilize both types of cores simultaneously), (ii) dynamic work partitioning (i.e., it distributes work unequally across threads), and (iii) dynamic power partitioning (i.e., it distributes power unequally across cores). This scheme analyzes the efficiency of the application threads running on the target heterogeneous multicore and redistributes the available chip power across cores to improve overall performance.

In Section 3, we examined multithreaded application workloads with a fixed number of threads, irrespective of their scalability. We then selected the best possible configuration from the total available cores at every stage of execution. Here we examine a general case, where applications can scale to different levels and demonstrate the need for heterogeneous architectures to cater to such a diversity of applications. We also examine the behavior of these applications under a fixed power budget, as a means to mitigate the problem of dark Silicon. We focus our attention on device-level heterogeneity in multicore processors and examine the viability of using a combination of CMOS-TFET based cores in processor design.

Fig. 4.5 EDP improvements as a result of migration to TFET cores under S1, S2, and S1+S2 strategies.
The ITRS roadmap [43] indicates that the maximum allowable chip power will remain constant over the next several generations even amidst technology scaling. This makes the chip power the most important parameter that constrains the maximum performance that can be extracted from a processor. Improving energy efficiency has become a major goal, which has lead to servers-on-chip with an entire power budget of a few Watts have been built for web applications as well as ‘big data’ applications like MapReduce and Hadoop [105]. The corresponding per-core power comes out to be less than 1W. Under such restricted power budgets, the number of active cores and their frequencies must be carefully determined to preserve efficiency.

4.3.2 Serving a Diversity of Applications

In order to efficiently map applications onto multicore processors under power constraints, two properties of the target applications must be studied: (1) scalability of the application with number of cores, and (2) scalability of the application with frequency.

In general, applications with a higher resource utilization in the cores see a higher benefit from increased core clock frequency. These application characteristics can be used to find better operating points (number of cores, frequencies) that improve performance under a given power budget. Consequently, different multithreaded applications can execute different operating points to maximize their performance under a power budget.

4.3.3 Exploiting Device-Level Heterogeneity

Figure 4.6 shows the average power consumption of the swim application when running on both CMOS and TFET-based homogeneous multicores with 8 and 32 cores. At the lowest frequency of 500 MHz, 32 TFET cores consume 3X less power than 32 CMOS cores. When
Fig. 4.6 Variation in CMOS and TFET processor power at different frequencies for different number of cores.

working under a fixed power budget, this can translate into more number of cores being turned on. Consequently, the massive parallel computational capacity brought in by 32 TFET cores can easily outperform the CMOS configuration when running a scalable application such as *swim*. At low voltages, it is also possible to operate the TFET at higher clock frequencies than CMOS. This enables the TFET cores to achieve superior performances than CMOS cores under power constraints for highly scalable applications.

When executing poorly scaling applications or sequential parts of parallel applications, it is better to restrict the entire power budget to a small number of cores running at a high operating frequency. The high-frequency optimized CMOS cores are very suitable for this operating point. Profiling simulations demonstrate that, given a power budget of 32W, only 8 CMOS cores can run at a peak frequency of 2 GHz. As a result, the 32 core heterogeneous CMOS-TFET architecture was designed with 8 CMOS cores and 24 TFET cores. This ensures that, when needed, 8 CMOS cores can be executed at the maximum frequency.
4.3.4 Porting Applications to Heterogeneous Multicores

After presenting alternative mapping and work partitioning schemes, an automatic scheme is proposed that can, without any programmer effort, achieve high performance, by answering the question: *How should the available power be partitioned across cores?*

4.3.5 Thread-to-Core Mapping

We first examine how the threads of a multi-threaded application should be mapped to the cores of a heterogeneous multicore. Considering a CMOS-TFET heterogeneous multicore, two possible methods are homogeneous mapping and heterogeneous mapping. Homogeneous mapping assumes that, while running a parallel region of an application, all threads will use the same type of core, i.e. it is executed either only on CMOS cores or only TFET on cores. This mapping strategy is used by traditional accelerator-based systems (e.g., a host processor with GPGPUs). However, the decision of what type of core leads to better performance under a fixed power budget must be made manually and is a non-trivial task.

An alternative way of mapping application threads to a heterogeneous processor is to run the application on both types of cores simultaneously. However, cores of different types will be executing at different operating points, which can cause significant performance discrepancy. Therefore, heterogeneous thread-to-core mapping can be expected to perform well only when used together with a heterogeneity-aware work partitioning scheme.

4.3.6 Work Partitioning

The simplest way of partitioning work across the threads of an application is to distribute the total work equally. This method is typically preferred in homogeneous systems where all
application threads will be executed on the same type of core. Running on the same core type results in all threads completing their computations at about the same time, minimizing the idle time of cores. On a loop-level parallelized system, static work partitioning distributes work statically at compile time, by assigning equally sized chunks of loop iterations to all threads.

In a heterogeneous multicore, the performance of threads running on different types of cores can be quite different under a heterogeneous thread-to-core mapping. Under static work partitioning, it can take some threads significantly more time than others to complete their computations, forcing the threads that finish early to wait idle for long periods of time. One solution is to use dynamic work partitioning, the most widely used form of which is dynamic loop scheduling [84]. In this scheme, iterations of a parallelized loop are not distributed equally across threads. Instead, each thread executes a different number of iterations proportional to its dynamic performance. As a result, all threads complete their allocated work at about the same time, maximizing the utilization of the parallel hardware.

4.3.7 Power Partitioning

As the total power is limited, it must be partitioned across cores such that the processor executes at an energy-efficient operating point. The optimum power distribution depends on the mapping and work partitioning policies adopted by the application. In case of homogeneous mapping and equal work partitioning, dividing power equally across all cores results in almost identical thread performance. However, for other mapping and work partitioning schemes, equal power partitioning may not be the best approach. Instead, adjusting the power distribution across cores can lead to more efficient operating points for the heterogeneous multicore.
Determining which type of core uses the available power more efficiently is application-dependent and can even change within an application based on the current phase of execution. Therefore, a runtime scheme that (1) identifies the energy efficiencies of different types of cores and (2) dynamically repartitions available power across cores to allocate more power to the efficient core type can potentially improve the overall performance. To improve performance of the CMOS-TFET heterogeneous multicore under a fixed power budget, a dynamic power partitioning algorithm based on a “perturb-and-observe” method is employed. Our algorithm considers the TFET cores and CMOS cores as two power domains and partitions the total chip power across these two domains. At every epoch, a fraction of the power is transferred one power domain to the other. At the end of the epoch, it is to be seen whether this action resulted in an improvement in the total performance in terms of instructions per second (IPS). If the performance improves, then the transfer of power is continued in the same direction; otherwise, it is reversed. Using such a dynamic scheme also enables us to react to behavioral variations occurring within the application. Our dynamic power partitioning algorithm takes the epoch length, percentage of power to transfer at each epoch, total available chip power, and the number of cores in either type as input parameters, and hence, is portable across various heterogeneities and power budgets. By employing power partitioning together with a heterogeneous mapping and dynamic loop scheduling, our goal is to reach higher performance automatically, without putting the burden of detailed analysis and testing on the programmer or the user.

4.3.8 Overview of Evaluated Schemes

Table 4.2 shows the schemes tested in our evaluation. Starting from a baseline homogeneous processor that is either all-CMOS or all-TFET with equal work partitioning and equal
power partitioning (CMOS-Base and TFET-Base), the first step we take is to switch to a 8-CMOS, 24-TFET heterogeneous processor. For each multi-threaded application, the programmer makes the binary decision of running parallel regions of the application on 8-CMOS or 24-TFET cores. Based on the ability of the programmer in making correct thread-to-core mapping decisions, the performance obtained varies. The schemes with the best and the worst mappings are referred to as Hetero-Manual-Best and Hetero-Manual-Worst, respectively. To explore the benefits of using all cores in the heterogeneous processor simultaneously, the Hetero-Simple scheme performs heterogeneous thread-to-core mapping. In the Hetero-DynWork scheme, on the other hand, the application is modified to be heterogeneous multicore-aware by adopting dynamic work partitioning. Our goal with this scheme is to evaluate the benefits of dynamic work partitioning on the heterogeneous system. Finally, the dynamic power partitioning scheme (Hetero-Auto) includes both heterogeneous mapping and dynamic work partitioning.

### 4.3.9 System Parameters and Simulation Infrastructure

### 4.3.10 Simulation Infrastructure

The Simics full system simulator was used for running our simulations [71]. The heterogeneous processor model comprised of 32 core system with 8 CMOS cores and 24 TFET cores.

<table>
<thead>
<tr>
<th>Processor</th>
<th>Mapping</th>
<th>Work Partitioning</th>
<th>Power Partitioning</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>32 CMOS cores</td>
<td>Homogeneous: CMOS</td>
<td>Equal</td>
<td>Equal</td>
<td>CMOS-Base</td>
</tr>
<tr>
<td>32 TFET cores</td>
<td>Homogeneous: TFET</td>
<td>Equal</td>
<td>Equal</td>
<td>TFET-Base</td>
</tr>
<tr>
<td>8 CMOS, 24 TFET</td>
<td>Homogeneous: CMOS or TFET</td>
<td>Equal</td>
<td>Equal</td>
<td>Hetero-Manual-Best</td>
</tr>
<tr>
<td></td>
<td>Homogeneous: CMOS or TFET</td>
<td>Equal</td>
<td>Equal</td>
<td>Hetero-Manual-Worst</td>
</tr>
<tr>
<td></td>
<td>Heterogeneous: CMOS and TFET</td>
<td>Dynamic</td>
<td>Equal</td>
<td>Hetero-DynWork</td>
</tr>
<tr>
<td></td>
<td>Heterogeneous: CMOS and TFET</td>
<td>Dynamic</td>
<td>Dynamic</td>
<td>Hetero-Auto</td>
</tr>
</tbody>
</table>

Table 4.2 Evaluated multicore configurations. The performance of these schemes are presented in Section 4.3.13.
The performance of this system is compared to a baseline system consisting of 32 cores (CMOS or TFET). The architectural parameters of the simulated systems are shown in Table 4.3.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>No. of Cores (Homogeneous)</td>
<td>32 CMOS or 32 TFET</td>
</tr>
<tr>
<td>No. of Cores (Heterogeneous)</td>
<td>8 CMOS and 24 TFET</td>
</tr>
<tr>
<td>L1 D/I-Cache</td>
<td>Private, 32KB each, 4-ways set-associ., 1-cycle latency</td>
</tr>
<tr>
<td>L2 Cache</td>
<td>Shared, 4MB, 16-ways set-associ. 10-cycles latency (2GHz)</td>
</tr>
<tr>
<td>Memory Access Latency</td>
<td>120-cycles (at 2GHz)</td>
</tr>
<tr>
<td>DVFS Epoch Size</td>
<td>1ms</td>
</tr>
<tr>
<td>Power Partitioning Epoch Size</td>
<td>5ms</td>
</tr>
<tr>
<td>Power Transfer Percentage</td>
<td>10%</td>
</tr>
</tbody>
</table>

Table 4.3 System parameters.

4.3.11 Dynamic Work and Power Partitioning Implementation

In order to eliminate the effects of load imbalance, guided parallel loop scheduling is used, which performs work partitioning at runtime. Here, the workload chunk size allocated to each thread is not fixed, but variable. The chunk size starts large but gradually reduces as more loop iterations are completed, which achieves a smaller scheduling overhead.

Power partitioning was incorporated as part of the simulation. At every epoch, the total number of instructions executed by all cores in the processor is calculated and the total number of instructions per second (IPS) is obtained. This calculation can be also implemented as an OS daemon process on a real system. Comparing the total IPS for the current and previous epochs, the daemon either decides to move power from the TFET domain to the CMOS domain or vice versa. Power re-distribution is carried at a period of 5 ms.

Our hardware DVFS implementation reads per-core power budget values from special purpose registers at every epoch. For each core, it determines the ratio of the power of the core to its power budget. It scales the voltage and frequency of the core such that it will match
its allocated power budget in the next epoch. Note that in our heterogeneous CMOS-TFET multicore, DVFS is carried out on a per-core basis so as to take the difference in the power and performance characteristics of different types of cores into account. Assuming a 50 mV/ns latency for on-chip per-core voltage regulators as presented by Kim et. al [50] and a PLL locking time of a few microseconds, the DVFS epoch length was set to be 1 ms. Selecting such a large DVFS epoch length, ensured that the performance overheads due to DVFS are negligible. DVFS levels are assigned at frequency intervals of 125 MHz, ranging from 500 MHz to 2 GHz.

4.3.12 Benchmarks

Based on the type of workload running on the heterogeneous multicore, the resource distribution problem is classified as:

- A multi-threaded application executing alone
- Two (or more) multi-threaded applications sharing the cores and power budget.

For both types of workloads, static and dynamic schemes were proposed to map the available cores and redistribute the available power to threads/applications on a heterogeneous multicore.

For the single-application, multithreaded workloads, applications are selected with suitable input sizes from the SPEC OMP 2001 suite [4]. These applications are parallelized using OpenMP [81] pragmas. The multi-programmed workloads were generated from random pairs of benchmarks from the Parsec suite. Each individual application is considered to be homogeneously mapped, i.e. all the threads of the same application are mapped to the same (CMOS or TFET) core type.
4.3.13 Experimental Results

To evaluate the performance of our 32-core heterogeneous processor with 8 CMOS and 24 TFET cores, a 32-core homogeneous CMOS and a 32-core homogeneous TFET processor were considered as baselines. A fixed power budget of 32W (i.e., 1W per core) was assumed in all evaluated configurations. In all configurations, sequential regions of the target applications are executed on CMOS cores, except with the homogeneous TFET system where the sequential regions are also executed on TFET cores. This core is then allocated the entire available power budget of 32W, in case the other cores are idle.


The performance of the heterogeneous multicore was compared against both homogeneous baseline processors, namely, a 32-core CMOS and a 32-core TFET multicore. Best-Base represents the maximum of CMOS-Base and TFET-Base performance. All results are normalized with respect to Best-Base. When comparing the proposed heterogeneous multicore against the baseline systems, the programmer is assumed to be capable of making perfect mapping decisions. In other words, the programmer analyzes each application and decides whether it is better to run the parallel regions of the application on the CMOS or TFET part of the heterogeneous processor. Making a bad thread mapping decision (e.g., mapping threads to CMOS cores although TFETs would perform better) can lead to sub-optimal performance. In Figure 4.7, the two programmer-directed mapping schemes are compared Hetero-Manual-Best and Hetero-Manual-Worst, which represent the best and worst mapping schemes for the target applications. It is observed that bad mapping decisions can degrade the heterogeneous system performance by 32% on average, which easily eliminates any benefit of using the heterogeneous system.
Therefore, making correct mapping decisions on heterogeneous systems is key to optimizing performance.

Fig. 4.7 Performance comparison of the best and worst thread mappings on our CMOS-TFET heterogeneous multicore.

4.3.15 Hetero-Auto vs. Hetero-Manual-Best

This dynamic scheme is performed automatically by a runtime system and requires no effort on the programmer’s part. Note that, while the programmer-directed mapping uses only one type of core at any time, our dynamic scheme uses both types of cores simultaneously. Further, applications running under our scheme also use dynamic work partitioning, whereas the manual scheme assumes a static, equal work partitioning.

Figure 4.8 shows our results with the two schemes, as well as two intermediate schemes. The Hetero-Simple scheme uses the heterogeneous CMOS-TFET processor with only heterogeneous mapping, i.e., it employs neither dynamic work partitioning nor dynamic power partitioning. We can see that, heterogeneous mapping alone makes our heterogeneous multi-core perform
Fig. 4.8 Performance of our CMOS-TFET multicore with and without dynamic work and power partitioning schemes.

4% better than the baseline. Adding dynamic work partitioning on top of heterogeneous mapping (Hetero-DynWork) brings an additional 12% performance improvement. On average, the dynamic power partitioning scheme (Hetero-Auto) performs 21% better than the best baseline performance.

In conclusion, superior sub-threshold characteristics of TFETs enable them to operate at points that cannot be realized by CMOS devices. Hence, by incorporating TFET devices, heterogeneous multicores can achieve higher performance, and serve a wider variety of applications. It is true that this improvement comes with an additional cost of using TFET technology and integrating CMOS and TFET technologies. However, the technology change to tri-gate transistors provided 18-37% improvements at the device-level [20], at a similar cost. The 21% system-level improvement presented in this work is of a similar order and can have significant ramifications in future processor design.
4.3.16 Runtime Characteristics

Figure 4.9 shows the dynamic variation in power consumption of *wupwise* application for the CMOS baseline (middle) and CMOS-TFET heterogeneous multicore (top), as well as the number of active cores (bottom). The total power consumption of both systems follow a similar trend: the entire power budget can be utilized in parallel regions, but sequential regions use only a fraction of available power as only a single core is active. The top figure also shows how our dynamic power partitioning scheme distributes the chip power across CMOS and TFET domains, which results in an improved overall performance.

![Figure 4.9 Power consumed by heterogeneous (top) and homogeneous (middle) systems, and number of active cores (bottom) over time (*wupwise*).](image)
4.3.17 Sensitivity Analysis

Here, the performance of the heterogeneous processor is analyzed under varying power budgets and ratios of CMOS and TFET cores.

- **Sensitivity to Power Budget:** Our results with the 8-CMOS, 24-TFET multicore with different power budgets for *wupwise* are shown in Figure 4.10. We observe that, for low power budgets, our heterogeneous multicore achieves a higher speedup. The reason for this behavior is that when power is scarce, cores in the homogeneous CMOS system are forced to operate at very inefficient operating conditions when compared to TFET cores employed in our heterogeneous processor. On the other hand, the homogeneous TFET processor still suffers from the limited maximum frequency of TFETs, which significantly degrades sequential region performance. In contrast, the heterogeneous multicore is able to utilize the available power more efficiently, leading to significant speedup. Since future processor trends point to a decrease in available power per core, the performance improvement obtained by the heterogeneous CMOS-TFET multicore is expected to increase even further.

- **Sensitivity to CMOS-TFET Core Ratio:** Figure 4.11 shows the variation in speedup with the number of CMOS cores in the 32 core processor. The two benchmarks demonstrated in this figure, namely, *wupwise* and *apsi*, show different trends as the CMOS core quantity is increased. 
  *wupwise* scales well with number of cores and prefers running at low frequencies on the largest possible number of TFET cores. Yet, a single CMOS core is needed to achieve high performance while executing the sequential regions of the application. Therefore, neither the CMOS-only nor the TFET-only baseline performs satisfactorily and achieves a performance improvement of up to 37% for the single CMOS core case. On the other hand, the scalability of *apsi* is limited, on
account of which it prefers running on a moderate number of CMOS cores. In this case, the best performance improvement of around 10% is observed with the 4 and 8 CMOS core systems. Having more or less number of CMOS cores results in a shift in the operating point of all CMOS cores and leads to a relatively inefficient execution.

We also experimented with different power transfer percentages and found that while amounts higher than 10% lead to oscillations, smaller amounts (large enough to trigger DVFS level changes) still perform satisfactorily.

![Performance of CMOS-TFET multicore under different power budgets](image)

**Fig. 4.10** Performance of CMOS-TFET multicore under different power budgets(*wupwise*).

### 4.3.18 Multiple Multi-threaded Applications Sharing a Heterogeneous Multicore

We propose static and dynamic optimizations to improve the performance of a power-constrained multicore when two applications are running concurrently. To simplify the problem, only a homogeneous application-to-core mapping is considered, where each application is assigned to either CMOS or TFET cores.

In the static scheme, the relative scalability of applications are first examined using static profiling. Working with two applications scheduled to run together, the application that scales
better with the number of cores is mapped to TFET cores and the application that scales better with frequency runs on CMOS cores. The total power budget is partitioned among the two applications (i.e., the CMOS and TFET domains) based on the ratio of the user-defined application weights.

Since power and core allocation in the profile-based scheme is fixed throughout the entire execution of the workload, it cannot capture the changing behavior of applications. Hence, the proposed dynamic scheme starts with the initial power allocation identified by the static scheme and dynamically repartitions power based on the energy efficiency applications achieve. In this case, since each application runs on only one type of core, the power allocated to each domain is distributed equally across its cores. To address fairness concerns, there is a limit of 10% on the maximum performance degradation that an application can suffer.

The configurations evaluated for this study are listed in Table 4.2. For the baseline, the best performing homogeneous configuration is chosen out of two 32-core processors with all-CMOS and all-TFET cores. when using static power partitioning (BestBase-StaticPow). The amount of power allocated to each application is decided statically based on the weights and the power budget. The number of cores to use for each application is selected using this
power allocation and profile-based scaling information. Each core uses DVFS while staying within the allocated per-core power budget. The heterogeneous equivalent of this static scheme (Hetero-StaticPow) considers what type of core would be better for each application when making application-to-core mapping decisions. The dynamic versions of these two schemes use the adaptive power partitioning discussed above (BestBase-DynPow and Hetero-DynPow).

![Graph showing weighted speedups](image)

**Fig. 4.12** Weighted speedups obtained with static and dynamic power partitioning methods when two applications are executed together (normalized to BestBase-StaticPow). Results are obtained for homogeneous and heterogeneous multicores under 40W (top) and 80W (bottom) power budgets. Application weights are given in parenthesis.

Figure 4.12 shows results obtained with two different power budgets (40W and 80W), normalized to the aforementioned baseline (BestBase-StaticPow). Dynamic power partitioning makes negligible impact on the homogeneous system as equal power partitioning is sufficient when all cores are identical. In the 40W case (top), the small per-core power budget results in the power transferred in the dynamic schemes being too small to cause transitions across DVFS.
levels. Therefore, these schemes yield very small benefits. Significant performance improvements are still observed due to the heterogeneous multicore design (25% with *Hetero-StaticPow* and 27% with *Hetero-DynPow*). On the other hand, for the 80W case, (bottom), the dynamic schemes yield higher speedups than the static schemes. This time, both the static and dynamic schemes enable the heterogeneous multicore to bring 13% and 21% performance improvement, respectively. The benefits obtained from the technology change reduces with increasing power as higher per-core power budgets reduce the advantage of TFET cores over CMOS cores.
Co-design of device and architecture in CMOS-TFET processors

5.1 Introduction

The previous chapter examined the tradeoffs between application scalability and core frequency in power constrained environments. However, in these experiments, all cores were assumed to have the same architecture configuration. Consequently, aspects involving architectural diversity and the effect of emerging device technologies on processor micro-architecture were yet to be considered. Hence, this chapter deals with co-designing systems by jointlycoupling technology and architecture-level heterogeneity. Designing processors with architecturally heterogeneous cores can result in an assymetric distribution of power across the processor. As a result, along with power limitations, thermal constraints also assume importance, especially in embedded domains. This is also due to the fact that the thermal limit directly impacts the product and packaging costs. These constraints affect overall form factor due to the need for additional cooling schemes and also diminish the energy efficiency. To this end, this chapter examines tuning architectural parameters like processor issue-width and frequency to operate under thermal constraints for various application domains. It also looks at employing cores that are heterogeneous in both device and architecture to maximize both the performance and energy efficiency. These various knobs present a multi-dimensional design space for diverse architectural domains ranging from the embedded space to the high-end server space. In an application domain such as mobile computing, constraints on peak temperature are especially stringent as there is limited
flexibility in terms of cooling techniques and longer wirelengths to reduce on-chip hotspots, as compared to higher end systems. For instance, cooling mechanisms are unaffordable due to the small form factor of the entire processor system, while increasing chip area significantly can increase die and manufacturing costs. In addition to microarchitecture components, I also examine the potential of extending the processor model to include customized accelerator core designs.

This chapter highlights the following contributions:

1. Using steep-slope device-based processors as complementary cores in systems like mobile processors that operate under tight thermal constraints. These processors would serve to expand the design space along with modifying architecture and system parameters, resulting in improvements to both performance and energy efficiency under these constraints.

2. Demonstrating techniques to optimally map single and multiple application workloads onto this heterogeneous mobile processor as well as techniques to dynamically swap application threads from one core-type to another, depending on the dynamic behavior of the application.

3. Re-examining the design of existing heterogeneous architectures such as the ARM big.LITTLE [33] processor when allied with device heterogeneity and conclude that a heterogeneous CMOS-TFET design can effectively run applications that prefer either a high operating frequency or wide-issue configurations that exploit high instruction level parallelism.

4. Stretching the boundaries of computing further into domains that are currently unreachable or difficult to achieve by traditional CMOS-based computers, including highly power-efficient customized acceleration units capable of low voltage operation.
5.2 Architectural considerations

Given fixed power and thermal constraints for a multicore architecture, there are several points in the design space that can be explored. From a microarchitectural perspective, this includes varying core complexity in terms of number of instructions fetched per cycle, issue width, size of register file and issue queue and the number of execution units. Depending on the microarchitecture configuration, the relative contributions of dynamic and leakage energy with respect to performance vary significantly. Further, workload characteristics also impact the efficiency of the various microarchitecture components. Based on the nature of the application, the impact on performance and energy due to the intrinsic datapath frequency or external resources such as the memory subsystem would also vary in different proportions.

Firstly, the diversity in application behavior across different device and microarchitecture configurations is examined. Figure 5.1 demonstrates the variation in energy-delay product (EDP)
for different core configurations for both Si FinFET and TFET based core designs. There is a wide variation across applications for the best core configuration that minimizes the EDP. In general, applications with high throughput (dedup, freqmine) are better able to exploit the higher core complexity on account of their higher instruction-level parallelism (ILP). On the other hand, applications like streamcluster show hardly any improvement with increase in issue width and would prefer operating more energy-efficiently on lower issue width cores.

5.2.1 Thermal constraints based microarchitecture design

Depending on the domain, the peak permissible temperatures vary. For instance, a mobile processor can tolerate a far lower peak temperature than a server. For instance, Samsung Galaxy phones containing ARMv6 processors are rated to operate at a maximum of less than 57°C (330K) [3], while most servers can attain upto 100°C (373K) temperatures. It is with this view that the possibility of particular architectural configurations for the thermally constrained mobile domain is examined.

The Hotspot-5.02 [38] thermal estimation tool was used for obtaining the peak core temperatures and generating thermal profiles. Since TFET devices share the same substrate and material characteristics as Si FinFETs apart from the few atoms used in doping, the thermal characteristics of TFETs are similar to that of CMOS devices. The CMOS and TFET power profiles used as input to the tool are obtained from periodic power traces using McPAT-1.0.
5.2.2 Frequency-complexity Tradeoffs

Since it is also possible to achieve higher processor complexities for a given thermal limit by simply reducing the processor frequency, a joint examination of the design space of both core frequency and processor complexity is done.

Fig. 5.2 Permissible states in the frequency-issue-width design space for CMOS and TFET processors at a) 330K, b) 340K and c) 350K temperature limits

Figures 5.2 shows the possible configurations that can be attained under different thermal budgets by CMOS and TFET processors. This figure clearly demonstrates that TFETs can operate at higher issue widths at lower temperatures, while CMOS cores can reach higher operating frequency as the thermal limit is increased.
Fig. 5.3 Comparison of thermal profiles of cores corresponding to the best performing CMOS configuration for temperature limits of a) 330K (1 issue, 1750MHz), b) 340K (2 issue, 2 GHz) and c) 350K (4 issue, 1.75GHz). CMOS frequencies below the crossover are not shown as TFETs are inherently more energy efficient at those points.

Figure 5.3 shows the variation in temperature across different microrarchitectural components in each processor configuration. The peak temperature of these cores determines which configuration is permissible under that thermal constraint. It is observed that, at lower thermal limits, CMOS cores have a very limited set of permissible microarchitectural configurations. In fact, even a single issue CMOS processor cannot operate at frequencies above 1.75 GHz for the 330K limit. On the other hand, TFET cores are able to operate at much higher issue widths. This compensates for the lower operating frequency of TFET processors as compared to CMOS. As the temperature limit exceeds 350K, CMOS cores are also able to operate at higher frequencies with sufficiently high issue width configurations. Since the benefits of increasing issue width beyond 4 in mobile applications are negligible, simulations restrict the peak issue width to 4. Consequently, the higher frequency of CMOS cores becomes the dominant factor and they outperform TFET cores.

In addition to performance, battery life is also a concern, especially for embedded devices. Hence it is necessary to minimize the energy under the previously described temperature
constraints. Based on these permissible configurations, it is possible to determine the best configuration in terms of both performance and energy.

5.3 Architecture Design Details

This section describes the techniques used to map applications on the embedded processor, under thermal constraints. In addition to a simple static mapping scheme, a dynamic mapping scheme based on the runtime instruction slack of the application is also proposed and evaluated.

5.3.1 Static mapping of applications

In the static evaluations, the best possible operating point is determined in the frequency-issue-width design space, for each application for different thermal limits. This gives rise to the configuration preferred by a majority of the applications for each temperature domain. Each application was run at its optimal frequency for that configuration. Depending on the static profiling results, it is possible to determine whether the application has a higher affinity for a CMOS or a TFET core.

Statically mapping applications to CMOS/TFET cores may not always achieve the desired results on account of periodic changes in program phase and characteristics. Hence, application phases with high ILP are run on TFET cores, which are capable of attaining more complex configurations than CMOS within the same thermal budget. However, their limited performance at high operating voltages precludes them from optimally running low ILP applications which prefer higher frequencies. In order to assign a metric to determine the degree of ILP of an application, a runtime slack estimation technique derived from [30] is used.
5.3.2 Slack-based dynamic mapping

Figure 5.4 shows the method used in estimating slack. Assuming no dependent instruction in the ROB, instruction $I_i$ can be delayed at most by $C_k$ number of cycles before the instruction $I_k$ is executed in its designated cycle. Without any dependencies across registers, all ready instructions could be committed in the same cycle. In this case, to estimate $C_k$ (in cycles), one would divide the number of instructions between $I_i$ and $I_k$ by the IPC of the current epoch ($IPC_{epoch}$). This is defined as $\Delta(i,k) = I_i - I_k$. This implies that instruction $I_i$ has enough slack until the exact cycle where $I_k$ gets committed. However, since only a finite number of instructions can be committed every cycle, instruction $I_i$ should be ready latest by $\Delta(i,k)/\max_{commit}$ cycles before $I_k$ is ready to commit. Thus the total slack in this case would be given by $C_k$, as shown. However, if instruction $I_j$ has a true dependency on instruction $I_i$, then instruction $I_i$ has to be executed before the instruction $I_j$, hence the slack in such a scenario would be denoted by $C_j$. 

![Fig. 5.4 Runtime Slack Estimation](image-url)
Most of the information required to compute slack is already in the processor. The dependency information for each instruction is stored in the ROB. The IPC can be estimated from the hardware performance counters.

Based on the slack estimated at runtime for each epoch, the application is set to run on a best configuration core based on the applications sensitivity to slack. If the slack determined in the epoch is higher than a prescribed threshold, it means that the application is less sensitive to slack. The application is then assigned to run on a low frequency high issue-width TFET core. Similarly, if the application is currently running on a TFET core and is very sensitive to slack, it is migrated to a high frequency CMOS core.

5.4 Simulation Infrastructure and Results

5.4.1 Simulation tools

Chapter 2 described the device simulations and the extracted Verilog-A model. The GEMS simulator [72] is used for performance estimation of each workload and is integrated with McPAT [61] to estimate the power consumption of the entire core as well as individual microarchitecture components. The power numbers obtained periodically from McPAT were then used by Hotspot-5.02 [38] to create a power trace and consequently a thermal profile of the core during the execution of the workloads.

5.4.2 Results: Static mapping of applications

Figures 5.5 and 5.6 show the speedup and energy of a static scheduling scheme, where the best core configuration is selected for each application. All results are normalized to a homogeneous system comprising of the best CMOS architectural configuration for that application.
It can be observed that the overall speedup and energy savings increases as the thermal limit is raised. This is because as the thermal budget increases, the number of attainable configurations in terms of issue width for the baseline CMOS core also increases. As a result, at the 350K limit, it is possible to operate a 4 issue CMOS core at a higher frequency than its TFET counterpart, negating any improvements on account of heterogeneity. The maximum harmonic mean speedup due to static mapping is observed to be 43% at 330K with an energy savings of 27%.

5.4.3 Dynamic migration of applications

When the dynamic migration scheme described in section 5.3 is implemented, it is possible to account for intra-application phases, thus further boosting the speedup and energy savings, Figure 5.7 and 5.8 show the speedup and energy of the dynamic scheduling scheme, DynMap. All results are normalized to a homogeneous system comprising of the best CMOS architectural configuration for that application.
DynMap outperforms the static scheme across most workloads. The largest improvement in performance and energy savings is seen in FFT. DynMap causes slight degradation in the performance of adpcm and gsm at 330K. Both these applications show high ILP as well as sensitivity to frequency. Consequently, migration to TFET, even for a few epochs, degrades performance significantly. DynMap outperforms the static scheme by 4%, 22% and 14% at 330K, 340K and 350K respectively. While DynMap is more energy efficient than static mapping at lower temperatures, consuming up to 10% lower energy at 340K, the energy for DynMap increases at 350K. This is because the TFET core operates at 1500 MHz, which is above $F_c$. Hence, the energy penalty for migrating to TFET cores is also large.

5.5 Enabling of emerging technologies on domain-specific architectures

It is evident that inherent physical limitations of the charge transport mechanism in MOS transistors have forced a rethink of several aspects of processor design, ranging from the device to the system level. In addition to general purpose processors, the study of computing with
Fig. 5.7 Speedup obtained on heterogeneous multicore due to DynMap w.r.t the best homogeneous CMOS configuration for thermal limits of 330K, 340K, 350K

Fig. 5.8 Normalized energy in heterogeneous multicore due to DynMap w.r.t the best homogeneous CMOS configuration for thermal limits of 330K, 340K, 350K
beyond-CMOS devices can be extended to domains that are currently unreachable or difficult to achieve by traditional CMOS-based computers, in particular the design of domain-specific accelerators with customized datapaths.

Figure 5.9 shows the block diagram corresponding to an accelerator for computing Euclidean distance between two vectors.

![Block diagram for computation of Euclidean distance](image)

**Fig. 5.9 Block diagram for computation of Euclidean distance**

As part of the experiments, a sample accelerator was synthesized, that computes the Euclidean distance between two vectors. This accelerator can be employed in feature-matching algorithms, where the Euclidean distance between a test feature descriptor vector (such as those
obtained from algorithms like SIFT or SURF) and every feature vector from the training database is computed for recognition tasks. The input consists of 64-dimensional vectors, each element being 16 bits in width. The accelerator consists of an 8-stage pipelined execution unit consisting of an array of multipliers and adders that compute the sum of squares of element-wise difference between a pair of vectors. The accelerator HDL code was simulated and verified using Synopsys VCS [98]. The designs were synthesized using the in-house TFET standard cell library, using Synopsys Design Compiler [99].

The overall execution time includes the data transfer time for streaming the input to the accelerator using DMA transfer from external memory and the accelerator computation time for 200 feature vectors in the training database and 1 test vector. For an input stream of images, the throughput of the TFET accelerator was computed to be 19392 frames/second.

The salient feature of TFET based accelerators is their capability to achieve high throughput with energy efficiency even while operating at low supply voltages. This is evident when the

![Fig. 5.10 Normalized delay, power, energy and EDP of TFET, iso-performance CMOS and iso-voltage CMOS accelerator designs. All results are normalized w.r.t the TFET design.](image)

The overall execution time includes the data transfer time for streaming the input to the accelerator using DMA transfer from external memory and the accelerator computation time for 200 feature vectors in the training database and 1 test vector. For an input stream of images, the throughput of the TFET accelerator was computed to be 19392 frames/second.

The salient feature of TFET based accelerators is their capability to achieve high throughput with energy efficiency even while operating at low supply voltages. This is evident when the
Fig. 5.11 Comparison of iso-performance, iso-voltage and peak-performance CMOS with TFET designs for a 32 point FFT accelerator

TFET-based Euclidean distance accelerator was compared with equivalent CMOS designs. An iso-voltage CMOS design operating at 0.3V is severely limited by its inherent sub-threshold operation and consequently has a transistor delay that is over 30X higher than TFET. Hence a TFET accelerator design is more feasible, even though its total power consumption is higher than the iso-voltage CMOS design due to its far superior switching speed. Consequently, the accelerator is much slower, and can be clocked at less than 100 MHz in order to meet timing constraints. In order to match the performance of the TFET accelerator, the CMOS design will have to operate at 0.54V. This increases the power and hence, the energy overheads. The performance, power, energy and energy-delay product results are summarized in Table 5.10.

These TFET-based designs are also compared with the peak-performing CMOS equivalent at 0.85V. For this purpose, a 32 point FFT accelerator design was synthesized using the Synopsys 32 nm library and scaled down to 22 nm by means of factors obtained from [28] and
ITRS projections, as shown in Figure 5.11. Although the critical path delay of the CMOS design (C4) is around half that of the TFET design (C1), its power consumption is over 30× of C1, which results in a huge energy and EDP advantage for the TFET-based design.

This chapter thus attempted to demonstrate the feasibility of device-architecture co-design in the context of manufacturing future processors. It described architectures tuned to specific embedded applications with stringent thermal and power constraints and proposed static and dynamic schemes to efficiently map representative applications onto these architectures. Finally, it also examined using steep-slope transistors for realizing highly energy efficient accelerator designs and demonstrated significant energy improvements over corresponding CMOS-based designs at different design points.
High Performance processors using Stacked 3D Tunnel FETs

6.1 Introduction

This chapter focuses on the impact that emerging devices and techniques will have on systems that are both plausible and preferable for mass deployment. In particular, it focuses on the rapidly maturing technique of 3D integration [14] and the potential benefits offered by designs built with Interband Heterojunction Tunnel Field Effect Transistors (TFETs).

Incorporation of 3D stacking technology into chip design has gained popularity due to the significant benefits that it offers, such as smaller package size due to reduced area footprint, decrease in communication power and delay on account of shorter wires, and increased bandwidth due to the higher availability of pins. While the viability of memory-on-memory and memory-on-core stacking has been demonstrated [66, 35], thermal and yield limitations have curtailed significant progress in this area.

Combining 3D integration and TFET designs offer the potential to extend the maximum number of aggressive cores possible on a single chip, within a viable yield and thermal budget. Yield decreases superlinearly with increase in area [14], and communication costs among cores scale poorly in planar designs [108]. Thus, 3D integration offers a very direct means to achieve meaningfully higher core counts in tightly integrated systems. However, moving to a 3D design aggravates thermal limitations by placing both additional heat sources and insulators between
the cooling system and lower layers in the stack. On the other hand, TFETs and other steep-slope devices offer fundamental reductions in leakage currents and switching energy at the cost of a more limited upper range of operating frequencies. There is a natural synergy between 3D integration and TFETs in that reducing the thermal density on each layer by substituting TFET designs for CMOS will allow more layers within the thermal budget, allowing 3D TFET based designs to scale to sufficient parallelism to overcome limitations in the serial performance of TFET based processors. However, while deploying TFET based designs in a 3D architecture is conceptually appealing, many questions regarding how best to design such a multiprocessor (e.g. microarchitecture selection, performance targeting, scaling) have to be addressed.

The points detailed in this chapter are as follows:

- An extensive evaluation of the performance and energy tradeoffs among choices in device technologies, 3D integration, microarchitecture, and scheduling under constraints imposed by realistic yield models, thermal bounds and exploitable application parallelism.

- An simulation-based demonstration of the viability of 3D integration of steep-slope based devices for achieving peak performance in highly parallel applications.

- Demonstration of the increase in the range of design space of applications that prefer steep slope technology-based cores, especially with technology scaling. This is also extended to show that the portion of the design space where CMOS is optimal shrinks to a point where only a small number of CMOS cores may be desirable.

- An intelligent scheduling approach for hybrid CMOS-TFET systems that allows less parallel applications to still achieve a significant fraction of their peak performance on a primarily TFET-based system.
6.2 Motivation

Broadly, performance improvements in general-purpose cores can be realized in two dimensions - by reducing single threaded latencies via increasing the frequency or by exploiting the inherent parallelism (TLP and ILP) of the application by increasing the number of application cores or architectural complexity (issue width, pipeline stages etc.). In theory, the increase in frequency can continue until fundamental physical properties of the transistors and wires allow it. Similarly, the increases from core counts are only restricted by the scalability of the application. However, in reality, there are several other constraints that crop up far earlier. Every processor is limited by the total power consumed, namely its power budget, which restricts the attainable processor configurations. This problem can be mitigated to an extent by various approaches [100] for exploiting Dark Silicon i.e by spatially or temporally reallocating power budgets such that either subsets of (possibly specialized) cores can operate at peak frequency or all cores can operate at peak frequency a subset of times at the expense of darkening/dimming other cores/times.

In addition to power, there are two other key considerations for understanding which processor configurations are practical. Namely, yield constraints may restrict the manufacturability of processors with high core counts [27] and thermal limitations due to power density may come into play even for processors staying within their aggregate power budget. These two constraints are examined in more detail and shows how emerging devices help us expand the design space in light of these constraints.
6.2.1 Thermal constraints on processor execution

Power budgeting has become an important consideration in the design and operation of processors. This power constrained operation can extend across a wide range of application domains, ranging from the mobile and embedded space to the high-end server space. However, constraining total power does not enforce adherence to the inherent thermal limitations of processor components. The component temperature depends, not on power, but on power density. Most processor components are rated to operate within a fixed range of temperatures and exceeding this temperature range can have an adverse impact on their lifetime and reliability. The Thermal Design Power or TDP is an indication of the peak power level that the processor can achieve without causing the thermal limit to be crossed.

6.2.2 Yield constraints on processor design

To exploit application parallelism, increasing per-chip core count can be done without significantly aggravating power density. However, increasing the die size can adversely affect the overall processor yield, since the yield is inversely proportional to the chip area. Folding cores onto multiple layers (e.g. 3D stacked chips) can reduce the area footprint. While this has ramifications both in increasing the processor yield as well as improving on-chip bandwidth and latency due to reduced interconnect length, there is a price to pay, in terms of bonding yield, for increasing the number of layers, and this limits returns on increasingly stacked chips. Further, increasing layer counts exacerbates thermal limitations, since the inner layers lack an efficient means for heat dissipation.
Figures 6.1a) and b) show the extent of frequency and core scaling for two applications, 
*barnes*, which scales well, and *ocean.cont*, which scales poorly. The regions shaded black corre-
spond to the points at which the scaling model “collapses”, i.e thermal and yield considerations 
restrict the design space. While both applications are affected by the frequency limitation, only 
*barnes* is adversely affected by the constraint on the number of cores.

**6.2.3 Opportunities with TFET processors**

As seen in the previous chapters, TFET cores can provide a more energy efficient alterna-
tive to conventional CMOS processors, especially at near-threshold and sub-threshold voltage –
at sufficiently low voltages, the steep slope of TFETs makes them inherently more efficient transistors independent of process tuning that can be done to customize CMOS. Substituting TFET cores for CMOS cores lessens the thermal consequences of 3D stacking. Consequently, stacked TFET cores extend the range of viable designs in the core count/frequency space. Similarly, operating CMOS cores at increased supply voltage ($V_{dd}$) enables high frequency operation.

There are several avenues to explore in order to trade-off the lower temperature operation of TFETs for increased performance. This chapter primarily focused on the advantages of extending device-level heterogeneity to 3D stacked processors, thus aiming to increase the design-space boundary illustrated in Figure 6.1. The two main roadblocks encountered in this effort are the decrease in yield due to bonding and TSV losses, and the steady increase in power density as layers are added, leading to large temperature increases among the internal layers.

### 6.3 Design Space Analysis of High Performance Systems

Table 6.1 describes the various dimensions evaluated in the 3D-device-heterogeneous architecture space.

**Table 6.1 Technology and System Parameters**

<table>
<thead>
<tr>
<th>CMOS Technology</th>
<th>22nm Si FinFET</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMOS Frequency Range</td>
<td>0.5 GHz - 3 GHz</td>
</tr>
<tr>
<td>TFET Frequency Range</td>
<td>0.5 GHz - 1.75 GHz</td>
</tr>
<tr>
<td>TFET Technology</td>
<td>22nm HTFET</td>
</tr>
<tr>
<td>Number of layers</td>
<td>1 - 8</td>
</tr>
<tr>
<td>Total Number of Cores</td>
<td>1 - 128</td>
</tr>
<tr>
<td>Number of utilizable cores</td>
<td>1-64</td>
</tr>
<tr>
<td>Thermal limit</td>
<td>360K (air cooled)</td>
</tr>
<tr>
<td>Ambient temperature</td>
<td>300K</td>
</tr>
</tbody>
</table>

This section analyzes each of the effect of each of these dimensions on the overall performance, under thermal and yield constraints.
6.3.1 Yield aware stacking of processors

As explained in Section 6.2, processor yield reduces super-linearly with increase in die area. As shown in equation 21 in [14], the yield varies with area as a Gamma function. While reducing the area footprint by stacking cores can improve the die yield, there are losses on account of joining 2 layers together, quantified by the bonding yield, as shown in equation 24 in [14]. As a result there is a tradeoff between increasing the die size and increasing the number of layers.

This yield variation for a multilayered processor system is counteracted by considering the use of redundant or spare cores. For a given multicore system, only a subset of the total cores on chip are assumed to be operational. The remaining cores are used to ensure that a minimum yield requirement of 50% is met. This is known as core sparing and is commonly used as a technique to improve the overall yield of several processors in industry [27]. Although additional hardware resources are spent on these spare cores, the improvement in yield significantly shortens the time to market for these processors. This is a far more viable alternative than aiming to improve the fabrication process both from a time and cost perspective. Adding a single spare core to an 8 core system can reduce the time to market by nearly 50%. However the number of spare cores needed to meet the yield criteria increases with the total number of cores. For larger number of cores and layers, the yield drops drastically, resulting in more than 50% of the cores being used for redundancy.

Figure 6.2 shows the number of stacked layers required to obtain a particular number of cores for different area footprints. The redundancy ratio is defined as the fraction of excess cores required to meet the yield threshold. It can be observed that both area footprint and number of
Fig. 6.2 Number of core layers required to realize a range of functioning cores for different area footprints. The fraction of redundant cores can be seen to increase both with area and with number of layers. Layers cause this redundancy ratio to increase. For smaller areas the reduction in yield due to bonding is a more dominant characteristic, as indicated by the increase in redundancy ratio for more stacked layers. However, as the area per layer increases, the yield decreases at a faster rate and folding the cores to stack them in multiple layers can arrest this decline. The maximum area footprint considered is 400 $mm^2$ per layer. In order to meet the yield constraint, it is essential to increase the number of layers to accommodate the redundant cores. This adversely affects the thermal behavior of the processor, further constraining the design space.

An important advantage that TFET technology has over other emerging devices is that it is compatible with the CMOS fabrication process [24]. Further, the process steps involved in the manufacture of TFET processors is similar to that of CMOS. Hence, this technology can be assumed to be similarly affected by process variation and displays similar yield as CMOS [89].
In order to account for uncertainties due to the new technology, yield experiments for TFET processors were run by reducing the baseline yield by 5% and 10%. The reduction in overall yield could be compensated by adding an additional redundancy of 8% and 17% respectively, without compromising the feasible design space for TFET processors.

Fig. 6.3 a) and b) Delineation of design space attainable by CMOS(red), TFET(blue), both (green) and neither (black) cores to obtain peak performance, for a scalable (barnes) and non-scalable (ocean.ncont) application respectively. The best performance is seen in the TFET configuration in barnes and in the CMOS configuration in ocean.ncont.

6.3.2 Modeling thermal distribution across multicores

In addition to affecting processor reliability and lifetime, the cost of cooling and packaging is determined by the thermal profile of the processor. Different cooling technologies such as microfluidic cooling can push this thermal limit up. For instance microfluidic cooling techniques for 3D processor-on-processor stacking can reduce core temperature by as much as
15°C [111]. For the purpose of this study, a thermal limit of around 85-90°C (358-363K) is used, assuming an air cooled machine. Microfluidic cooling could enable the temperature bound to be raised to around 100-105°C.

6.3.3 Variation in microarchitecture

The ability to exploit the greater microarchitecture complexity or increased number of cores depend on the application characteristics, in particular the ILP and TLP of the application. As discussed in Section 6.2, there is a region in the parallelism v/s frequency plot that is not attainable because of yield and thermal constraints. By using a combination of power-efficient TFET technology with high performance CMOS can expand the design space. Using TFET cores in conjunction with 3D technology can reduce the power consumed, while maintaining processor yield, thus mitigating the thermal constraint. However, whether this extra design space manifests itself as a performance improvement depends entirely on the application scaling behavior.

Figure 6.3a) and b) show the 2 applications, barnes and ocean.cont, respectively, that represent the extreme edges of application scaling with cores. The additional TFET cores operating at low frequency, would prove beneficial for highly parallel applications like barnes which is then able to improve its peak performance, as seen in Figure 6.3a). On the other hand, an application like ocean.ncont, which has limited TLP, prefers operating on fewer cores at higher frequency, as shown in Figure 6.3b). It is evident that barnes benefits greatly from the extra number of cores, whereas the effect is not very significant in ocean. ncont.
6.4 System Infrastructure

Simulations were carried out on a multicore system comprising primarily of Intel Ivybridge-like cores. Table 6.2 lists the system configuration for the default simulations.

Table 6.2 Core and memory configurations

<table>
<thead>
<tr>
<th>Processor</th>
<th>Ivybridge Microarchitecture</th>
</tr>
</thead>
<tbody>
<tr>
<td>L1 Cache</td>
<td>32 KB D/I, 8 way S.A</td>
</tr>
<tr>
<td>L2 Cache</td>
<td>256 KB private Cache</td>
</tr>
<tr>
<td>L3 Cache</td>
<td>2MB/Core shared LLC, 16 way S.A</td>
</tr>
<tr>
<td>DRAM</td>
<td>4GB, DDR3-1600, 1 mem channel</td>
</tr>
</tbody>
</table>

6.4.1 Architectural simulation setup and benchmarks

The Sniper-5.0 [13] system simulation tool was used for performance simulations. This tool is integrated with McPAT-0.8 [61] which is used for power and area estimation. The McPAT technology file was instrumented with parameters obtained from the TCAD simulations. These parameters are listed in Table 6.3. In addition McPAT interfaces with Cacti, providing timing models for every processor component as well as wires, which are used to determine the critical path delay.

Table 6.3 Technology parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>FinFET</th>
<th>HTFET</th>
<th>Parameter</th>
<th>FinFET</th>
<th>HTFET</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_{ox}(fF/\mu m)$</td>
<td>1.28</td>
<td>1.28</td>
<td>$L_g$(nm)</td>
<td>26</td>
<td>26</td>
</tr>
<tr>
<td>$V_{th}$(V)</td>
<td>0.25-0.3</td>
<td>0.1 (eff)</td>
<td>$V_{d-sat}$(V)</td>
<td>0.419</td>
<td>0.288</td>
</tr>
<tr>
<td>$R_{on}(K\Omega-\mu m)$</td>
<td>1.01</td>
<td>2.43</td>
<td>$I_m$(mA-\mu m)</td>
<td>0.71</td>
<td>0.166</td>
</tr>
<tr>
<td>$C_{g-ideal}(fF/\mu m)$</td>
<td>0.55</td>
<td>0.327</td>
<td>EOT(nm)</td>
<td>0.7</td>
<td>0.7</td>
</tr>
<tr>
<td>Source</td>
<td>1e20</td>
<td>4e19</td>
<td>Drain</td>
<td>1e20</td>
<td>8e17</td>
</tr>
<tr>
<td>Doping(/cm$^3$)</td>
<td>n+</td>
<td>GaSb p+</td>
<td>Doping(/cm$^3$)</td>
<td>n+</td>
<td>InAs n+</td>
</tr>
</tbody>
</table>

For the purpose of obtaining thermal profiles, periodic traces were created using Sniper and created a power profile by running McPAT on each individual trace. The processor logic
and wire models, described in Section 6.3 were used to obtain the corresponding TFET numbers from the CMOS core simulations. These power profiles were then used as input to Hotspot3D for obtaining temperature variations across the processor.

6.4.2 Modeling thermal variation

For determining the thermal distribution across a 3D stacked multicore, the Hotspot-5.02 tool [38] was used. As shown in [11], the power budget of a multicore is based on its thermal profile, which in turn, depends on the temperature (or power) distribution across adjacent cores. This is done in order to take into account the effect of heat dissipation across core boundaries. Hence, this phenomenon was analyzed by running simulations on a system of 3 cores arranged side-by-side, each one operating on a similar workload. The central core will naturally experience the highest temperature distribution, since the surrounding cores operating at high temperature offer limited avenues for heat dissipation. Secondary effects beyond multiple core boundaries are negligible and can hence be ignored. Figure 6.4 demonstrates the variation in thermal behavior of three such 4-issue CMOS cores running at 2 GHz. The central cores in each layer have a larger area of peak temperature due to its higher power density as compared to the cores at the boundary. This model can be replicated to correspond to several cores on the same layer. An 8 layered multicore with a (20 µm) thin thermal insulating material between each core layer was considered as the 3D stacked design, with each layer comprising of a multitude of such cores. The additional temperature increase due to transition to 3D is modeled in Hotspot3D [44].
6.4.3 Scheduling diverse workloads on a stacked CMOS-TFET multicore

To demonstrate the motivation behind using a heterogeneous 3D stacked multicore for efficiently executing a diversity of applications, a configuration comprised of a single (top) layer of CMOS cores and remaining layers of TFET cores is used. In order to carry out a comprehensive study of the different workload characteristics encountered by the system, each benchmark was statically profiled and their thread-level parallelism and memory utilization was determined. A selection of diverse multiprogrammed workloads are created from the Parsec, Splash2 and SPEC CPU2006 suites for this purpose, based on these characteristics. The benchmarks are characterized as the following:

1. Multithreaded - High scalability

2. Multithreaded - Limited scalability

3. Single-threaded (no scalability) with high memory utilization

4. Single-threaded (no scalability) with low memory utilization
Figure 6.5 shows the benchmarks that were profiled and used for obtaining representative workloads.

By randomly combining pairs of workloads from these categories, several distinct classes of multiprogrammed workloads are obtained. The thermal constraint under which the multicore configuration operates primarily holds when majority of cores are active. On the other hand, the shared L3 cache, having a much smaller activity factor does not consume as much power as cores and is consequently much cooler. This make it possible to operate the on-chip caches when majority of cores are turned off, during execution of single threaded applications. Thus, depending on the workload characteristics, the system can either be operated as a 3D stacked processor multicore or as a single (or multiple) layer of cores with a large stacked L3 cache.
Figure 6.6 illustrates the proposed heterogeneous configuration and the possible states it can operate under for different workloads. The assignment of cores is as shown in the figure. Depending on the memory utilization as determined previously, the cache is partitioned according to the following heuristic, termed as *Heterogeneity Aware Scheduler (HAS)*.

- The L3 cache local to the core is initially allocated to the application running on that core.
- The remaining L3 cache local to unused cores are preferentially allocated to applications depending on whether they are sensitive to cache size or not, as determined by their *MPKI* (Misses per Kilo Instruction).
- If both applications have high cache utilization, then the un-allocated L3 cache is partitioned equally between applications.
- If both applications have poor cache utilization, then the un-allocated L3 cache is left unused, in order to preserve locality and reduce access latency.
- If the applications have differing L3 cache utilization characteristics, the application with higher utilization is allocated the unused cache.

Based on cache utilization, applications are classified as either dependent or independent of cache size without finer grained comparisons of relative utilization between them. This is because the true working set size of an application can be highly data dependent and the response of the application to increasing or decreasing the cache size may not be deterministic.

Thus, an algorithm is proposed, that optimally utilizes the on-chip resources for a wide variety of applications to maximize the thermally constrained performance. Each application is profiled *a priori* to determine its scalability and memory utilization.
Fig. 6.6 Different operating states of the heterogeneous multicore: a) 2 highly scalable parallel applications scheduled on the entire multicore. b) 2 completely sequential applications scheduled exclusively on CMOS cores. c) A sequential application, scheduled on a CMOS core, running alongside a weakly scaling application. The former is scheduled on a single CMOS core, while the latter is scheduled on either the remaining CMOS cores or TFET cores depending its optimal configuration. d) A sequential application, scheduled on a CMOS core running alongside a highly parallel application, scheduled on the entire set of TFET cores.

A combination of workloads was selected from the Splash2, Parsec [10] and SPEC CPU2006 suites. For the heterogeneous scheduling experiments, random combinations of workloads with different scaling and memory utilization characteristics are used, as described in Section 6.3. Table 6.4 shows the workload mixes that were evaluated.

6.5 Results

This section attempts to find the best possible device-architecture co-design for a diversity of workloads. Studies were carried out by varying the core frequency and the number of cores across multiple stacked layers, under the thermal and yield constraints described in
Table 6.4 Configuration of the evaluation platform.

<table>
<thead>
<tr>
<th>Workload-mix</th>
<th>W1 characteristic</th>
<th>W2 characteristic</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Scaling, MPKI</td>
<td>Scaling, MPKI</td>
</tr>
<tr>
<td>mcf-gobmk</td>
<td>No, high</td>
<td>No, high</td>
</tr>
<tr>
<td>lbm-scluster</td>
<td>No, high</td>
<td>Weakly, high</td>
</tr>
<tr>
<td>mcf-canneal</td>
<td>No, high</td>
<td>Strongly, high</td>
</tr>
<tr>
<td>gcc-sphinx</td>
<td>No, low</td>
<td>No, high</td>
</tr>
<tr>
<td>barnes-fanim</td>
<td>Strongly, low</td>
<td>Strongly, low</td>
</tr>
<tr>
<td>ocean.nc-raytrace</td>
<td>Weakly, high</td>
<td>Strongly, low</td>
</tr>
<tr>
<td>ocean.sc-scluster</td>
<td>Weakly, high</td>
<td>Weakly, high</td>
</tr>
<tr>
<td>canneal-ocean.nccont</td>
<td>Strongly, high</td>
<td>Weakly, high</td>
</tr>
</tbody>
</table>

Section 6.3. They also demonstrate sensitivity results over a range of temperature budgets and microarchitectural configurations.

6.5.1 Optimal operating points in the design space

Figure 6.7a) and b) show the various optimal design points that are possible for different sets of applications (parsec and splash2 respectively). The harmonic mean of the relative speedups of all applications in each benchmark suite (relative to a single core, operating at peak frequency), at every operating point is evaluated for each category. In addition to the red and blue bars, which signify the operating points exclusive to CMOS and TFET cores respectively, the green colored bars denote all states that can be attained by both core types. The diversity in the overall scalability of the workload suite is evident in the comparison between Figures 6.7a) and 6.7 b). In order to determine which core configuration is preferred in the green region, the CMOS and TFET power is calculated for all states in this region and plot the power savings obtained by using one core over the other, as shown in Figure 6.8. In this figure, the red and blue regions correspond to those core states where it is more power efficient to use CMOS or
Fig. 6.7 a) and b) Mean speedup of different applications in the splash2 and parsec suites respectively. The splash2 applications, on average, prefer higher frequency and fewer cores to operate. On the other hand, parsec benchmarks operate most efficiently on larger number of cores and lower frequencies, with only $\sim 17\%$ applications preferring high frequency CMOS cores as compared to 29$\%$ of splash2 applications.

Fig. 6.8 a) and b) Mean power savings obtained in parsec and splash2 suites respectively by running workloads on CMOS (red region) or TFET (blue region) cores. This plot is restricted to the region where both CMOS and TFET cores are operable in order to determine the more efficient core.
TFET respectively. This plot clearly illustrates that for optimal performing designs in the TFET-preferred region, the power savings can be significant and as shown in Figure 3.9, the relative gains w.r.t CMOS will increase with subsequent generations.

Figure 6.9 shows the performance comparison of a 4-issue Ivybridge TFET v/s CMOS processor for a range of Splash2 and Parsec benchmarks and compares the best performing configurations in each case. The optimal configurations for each processor (frequency, number of layers), subject to thermal and yield constraints are indicated for each data point. All speedups are normalized to a single CMOS core running at peak frequency (3 GHz). The TFET core configurations outperform the best CMOS configuration by an average of around 17% for the Splash2 suite and around 20% for the Parsec suite. The overall speedup is around 18%. This performance improvement varies with the temperature budget as shown below. Table 6.5 shows the best performing configuration under thermal constraints in terms of frequency, number of cores and number of stacked layers for both CMOS and TFET processors.

Fig. 6.9 Relative performances of 3D stacked CMOS and TFET configurations using an 8 stacked layers consisting of 64 functioning 4 issue processors. The thermal budget assumed here is 87°C.
Fig. 6.10 Variation of performance improvement of TFET core as opposed to CMOS cores for different thermal limits. Evaluations are carried out separately for Splash and PARSEC benchmark suites.

Table 6.5 Best performing configuration for each workload

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>Technology</th>
<th>Frequency(GHz)</th>
<th>Cores</th>
<th>Layers</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SPLASH</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>barnes</td>
<td>CMOS</td>
<td>1</td>
<td>64</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>TFET</td>
<td>1.25</td>
<td>64</td>
<td>8</td>
</tr>
<tr>
<td>fmm</td>
<td>CMOS</td>
<td>1</td>
<td>64</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>TFET</td>
<td>1.75</td>
<td>64</td>
<td>8</td>
</tr>
<tr>
<td>ocean.cont</td>
<td>CMOS</td>
<td>1.75</td>
<td>32</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>TFET</td>
<td>1.5</td>
<td>32</td>
<td>4</td>
</tr>
<tr>
<td>ocean.ncont</td>
<td>CMOS</td>
<td>1.75</td>
<td>32</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>TFET</td>
<td>1.5</td>
<td>32</td>
<td>4</td>
</tr>
<tr>
<td>radiosity</td>
<td>CMOS</td>
<td>1</td>
<td>64</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>TFET</td>
<td>1.25</td>
<td>64</td>
<td>8</td>
</tr>
<tr>
<td>water.nsq</td>
<td>CMOS</td>
<td>1.5</td>
<td>32</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>TFET</td>
<td>1.25</td>
<td>64</td>
<td>8</td>
</tr>
<tr>
<td>water.sp</td>
<td>CMOS</td>
<td>1</td>
<td>64</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>TFET</td>
<td>1.25</td>
<td>64</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>PARSEC</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>blackscholes</td>
<td>CMOS</td>
<td>1</td>
<td>64</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>TFET</td>
<td>1.25</td>
<td>64</td>
<td>8</td>
</tr>
<tr>
<td>canneal</td>
<td>CMOS</td>
<td>1</td>
<td>64</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>TFET</td>
<td>1.25</td>
<td>64</td>
<td>8</td>
</tr>
<tr>
<td>fanimate</td>
<td>CMOS</td>
<td>1</td>
<td>64</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>TFET</td>
<td>1.25</td>
<td>64</td>
<td>8</td>
</tr>
<tr>
<td>raytrace</td>
<td>CMOS</td>
<td>1</td>
<td>64</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>TFET</td>
<td>1.25</td>
<td>64</td>
<td>8</td>
</tr>
<tr>
<td>scluster</td>
<td>CMOS</td>
<td>1.75</td>
<td>32</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>TFET</td>
<td>1.5</td>
<td>32</td>
<td>4</td>
</tr>
<tr>
<td>swaptions</td>
<td>CMOS</td>
<td>1</td>
<td>64</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>TFET</td>
<td>1.25</td>
<td>64</td>
<td>8</td>
</tr>
</tbody>
</table>
6.5.2 Sensitivity to thermal budget

Figure 6.10 shows the variation in performance improvement obtained by TFET by comparing the best possible TFET and CMOS core configurations. TFET cores are clearly the preferred choice for thermal budgets up to around 360K (87°C), while the performance difference is negligible up to around 380K (107°C). At higher thermal budgets above a 100°C, CMOS cores clearly dominate since, the scope of microarchitectural configurations that they can attain is large enough to offset the increased thermal efficiency of TFET cores.

6.5.3 Sensitivity to microarchitecture

Experiments were carried out, that compare CMOS and TFET performance for a host of processor microarchitectures, ranging from a single issue to an 8-issue out-of-order processor, as shown in Figure 6.11. This figure shows the mean speedup of all benchmarks run from the Splash2 and Parsec suites, when compared to a single core baseline running at peak frequency (3 GHz). The 3D stacked multicore configuration remains the same as the previous experiments and is subjected to the same thermal limit of 360K. The Splash suite of benchmarks are not very sensitive to processor complexity as there is only a minor improvement in speedup with increase in issue width. On the other hand, in case of the Parsec suite, the performance improvement of TFET processors peaks at the 4 issue configuration. This is because the 4-issue TFET processor has sufficient capacity to exploit the inherent ILP of the application. As a result, when combined with 3D stacking, this configuration is able to extract the maximum performance from the application by optimizing both its ILP and TLP. For lower issue processors, core frequency plays a more important role, which reduces the advantage due to TFET cores. On the other hand, wider (6 and 8 issue) processors are extremely power hungry and provide limited improvement in ILP
over the 4 issue configurations. As a result, the higher base temperature attained by these cores, severely limits the microarchitectural flexibility in both CMOS and TFET cores, leading to lower speedups.

**Fig. 6.11 Comparison of performance speedup of CMOS and TFET cores for different microarchitectural configurations. Evaluations are carried out separately for Splash2 and Parsec benchmark suites for issue widths of 1 to 8.**

### 6.5.4 Heterogeneity aware scheduling

In addition to the static profiling based results, the viability of a stacked CMOS-TFET heterogeneous multicore was also demonstrated. A *Heterogeneity-Aware Scheduler* was implemented by running the workload mixes in 6.4 on the CMOS-TFET multicore and improvement in performance was compared to both a homogeneous CMOS and a homogeneous TFET multicore. The results are shown in Figure 6.12. All results are weighted speedups normalized to the ideal baseline, i.e the weighted speedup of each application when run individually on the best possible CMOS/TFET configuration. The heterogeneity aware scheduler results in a 17% improvement over the best homogeneous configuration.
In conclusion, this chapter demonstrates an extensive evaluation of the performance and energy tradeoffs among choices in device technologies, 3D integration, microarchitecture, and scheduling under constraints imposed by realistic yield models, thermal bounds and exploitable application parallelism. It showed that, with 3D integration, steep-slope based devices can extend the space of viable designs and achieve competitive performance for highly parallel applications. It highlights the fact that, with further technology scaling, the range of applications for which steep-slope devices are appropriate grows, while the portion of the design space where CMOS is optimal shrinks to a point where only a small number of CMOS cores may be desirable. Finally, a method was demonstrated for capitalizing on these trends by developing an intelligent scheduling approach for hybrid CMOS-TFET systems. This method was shown to yield mean improvements ranging from 19% for a high end server workload running at around 90°C to over 160% for embedded systems running below 60°C.
Chapter 7

Design Space Exploration of
Workload-Specific Last-Level Caches

The previous chapters explored the various ways that the design of cores is impacted by the introduction of new device technologies. Here, this study has been extended to the memory hierarchy. While there have been several works on finding alternatives to main memory from among different emerging non-volatile memories [58, 55], I shall focus on exploring the various technologies that could be useful for designing on-chip last-level caches. I shall also examine the various energy-saving architecture techniques that could be allied with these new device technologies. Thus, in this chapter, I shall investigate different technologies that are feasible for memory design as well as the various architectural techniques used for maximizing the power efficiency in on-chip last-level caches for embedded systems. Studies have been carried out for a variety of application domains, ranging from ultra-low power sensing applications to SoC-based systems with a moderate power constraint.

The main contributions of this chapter are:

- Modeling various cache designs and their effects on system performance and energy using full-system simulations. A system-level evaluation is performed for three configurations, each configuration representing a different usage scenario (ultra-low, low, and medium energy).

- Evaluating energy, performance, EDP, and ED² numbers for iso-area last-level cache designs.
• Evaluating private/shared and homogeneous/heterogeneous caches for dual-core processors running multiprogrammed workloads.

7.1 Energy-efficient Cache Implementations

This section presents various cache designs proposed to reduce energy consumption in last-level caches. These caches either are implemented using new technologies or employ special energy reduction techniques. Although our evaluation has been done on a 2-level cache system, it can be easily extended to deeper cache hierarchies since the underlying considerations remain the same.

7.1.1 Baseline: HP CMOS SRAM

High performance (HP) CMOS transistors can deliver large drive currents (as high as 1628 uA/um for a 22nm FinFET [43]), which makes them preferred in implementing latency-critical components such as cores. The disadvantage of HP CMOS transistors is that they have a large off-current (in the order of 100nA/um) which contributes to the high leakage power of the device.

Caches built using HP CMOS transistors typically employ the standard 6T SRAM cells. Large drive currents provided by these devices enable low cache access latencies. However, even when the SRAM cells in the cache are idle, the large leakage current passing continuously through a large number of transistors in a large cache can result in very high leakage power.
7.1.2 Voltage Scaled HP CMOS SRAM

The supply voltage of HP CMOS caches can be statically set to a lower value than the nominal voltage to reduce power consumption [106]. Reducing supply voltage brings both leakage and dynamic power savings, yet these savings come with the cost of reduced drive current which increases cache access latency. Near-threshold caches can lead to significant reduction in power consumption, but the associated latency can be prohibitive.

7.1.3 LSTP CMOS SRAM

Threshold voltage is a very important factor in determining the off-current of a transistor. Low stand-by power (LSTP) devices [43] are designed to have much higher threshold voltages (0.28V in HP vs. 0.63V in LSTP) which results in about 4 orders of magnitude reduction in off-current (from 100nA/um to 10pA/um). By replacing the leaky HP transistors in the cells of an SRAM cache with low leakage LSTP transistors, the cache leakage can be significantly reduced. However, this increase in threshold voltage also reduces the on-current of the transistor, which leads to an increase in cache access latency as well as the dynamic energy. LSTP devices also require additional process steps to manufacture transistors due to their increased threshold voltage.

7.1.4 TFET SRAM

When power becomes a significant concern, Inter Band Tunnel-FETs (TFETs) [77] can be considered as a technology replacement. TFET devices show much higher energy efficiencies than CMOS devices below 0.5V [2]. The benefits of TFET devices come from the fact that at low voltages they can provide higher drive currents than CMOS devices. Further, TFETs show
extremely low leakage, which means that they can be good candidates to implement caches in low power applications. TFET technology is compatible with the CMOS process flow and requires a small number of additional steps in the flow. Among emerging technologies, the process compatibility of TFETs is considered to be superior to others. On account of their low operating voltages, TFET-based SRAMs are more susceptible to variation-induced errors. There are several existing variability mitigation techniques for 6T CMOS SRAM cells operating at near-threshold and sub-threshold voltages [110][53]. However, these techniques do not work well for the corresponding TFET-based designs. [89] discusses the design of a variability-resilient 10T TFET SRAM cell. However, the increase in the number of transistors in the TFET SRAM cell results in a reduction in capacity. Under the same area, a variation-aware 10T TFET SRAM cache can realize about 67% of the capacity of a CMOS SRAM cache.

### 7.1.5 STT-RAM

STT-RAM is a magnetic storage technology where the spin-transfer torque (STT) effect is used to invert the data stored in the magnetic storage element [37, 46]. Typical STT-RAM cells employ the standard 1T-1MTJ structure, where the polarity of the magnetic tunnel junction (MTJ) represents the stored data. Writing into an STT-RAM cell requires applying a strong and long current pulse that re-aligns the free layer in the MTJ in the opposite direction. This results in STT-RAM caches suffering from high write energy and write latency. Currently, typical values for the magnitude and duration of this write pulse are 50-100 uA and 10-35 ns [43, 25].

Although the underlying 1T-MTJ cell structure in STT-RAMS does not suffer from leakage current, the leakage power of the peripheral circuitry (write amplifiers etc.) is much more than that of HP CMOS caches. This is because, the peripherals must be sized bigger to deliver
the large currents needed to write into the STT-RAM cell. Another disadvantage of STT-RAM is that, although it is considered to be compatible with CMOS process flow, it still requires an additional 3-4 masks [2]. This increases the chip manufacturing cost.

There are some parameters to be considered in the design of STT-RAM caches, such as the organization of the subarrays and the sizing of the peripherals. By tuning these parameters, STT-RAM caches can be optimized for access latency or capacity. Capacity-optimized STT-RAM caches can have up to 4x capacity of HP CMOS under the same area. However, optimizing for capacity results in high access latencies which can penalize performance. Alternatively, latency-optimized STT-RAM caches have lower access latencies, but they cannot realize capacities as high as the capacity-optimized STT-RAM cache.

### 7.1.6 Power Gated/Drowsy HP CMOS SRAM

Power gating [79] and drowsy modes [31] are two methods used to reduce the leakage power of HP CMOS caches. In power gating, an idle cache line is put to sleep by disconnecting the cells from the power supply. Although the cache line suffers almost zero leakage power, its contents get lost. In the drowsy mode, the supply voltage is reduced to a level that can barely keep the stored data [85]. While this does not eliminate cell leakage entirely, it can reduce it significantly. When a block is put into drowsy mode, the data it contains is not lost, but an access to this cache line requires the supply voltage to be increased to the full level before serving the request. Power gating and drowsy caching techniques require high threshold power gates to block current flow into the cells, which would increase the area overhead.
7.2 Simulation infrastructure and Application domains

This section presents the simulation infrastructure, the systems/workloads simulated, and the metrics used to compare the caches discussed in Section 7.1.

7.2.1 Core and Cache Modeling

A simple in-order core model was assumed and its energy was calculated using methods described in Chapter 3. In addition cache voltage scaling was also implemented to model CMOS SRAM caches operating at reduced voltages (0.7, 0.6, and 0.5V). For power gated and drowsy caches, three sleep intervals were used (100, 100K, 10M cycles). As a result of these simulations, the capacity, leakage power, dynamic energy per access, and access latency values were obtained for all of target caches. A list of caches evaluated in this work is given in Table 7.2.

7.2.2 Simulation Infrastructure

Using the core and cache latency/energy numbers obtained using the above methods, simulations were carried out using the Simics full-system simulator. To represent different usage scenarios, the following three configurations were considered:

- **Ultra-low Energy Domain**: This domain represents the scenario where the total available energy is extremely scarce and the battery lifetime is very critical. The available power budget is assumed to be less than 100mW. Systems used for wireless sensing and some embedded applications can be considered as examples.

- **Low Energy Domain**: This domain represents the scenario where energy is important, but performance cannot be sacrificed indefinitely for lower energy. This domain is assumed to
operate within a power constraint of around 500mW. Most embedded applications (e.g., handheld devices whose battery can be recharged, but improved lifetime is useful) fall into this domain.

- **Medium Energy Domain**: This domain represents the scenario where performance becomes more important than improvements in energy. Consequently the operating power of the core and cache is assumed to be around 1-2W. Some performance-critical embedded applications and low-end laptop computers are in this category.

For each domain, a different configuration and a different set of applications were used to better capture its needs and characteristics. Details of our target configurations are shown in Table 7.1. A latency-optimized HP-CMOS L1 cache and a capacity-optimized off-chip DRAM to be constant across all cache configurations and power domains. Experiments were also carried out with dual-core systems running multiprogrammed workloads. The dual-core configuration space also included shared/private and homogeneous/heterogeneous caches. Further, the effect of uneven distribution of area across private L2 caches of different cores on overall system performance and energy was also analyzed.

### 7.2.3 Evaluation Metrics

The criteria for efficient execution can differ widely across energy domains. While performance may be compromised on platforms with a strict limit on available energy, other platforms may expect the highest performance with little emphasis on the power consumed. In order to make domain-specific comparisons, the ultra-low energy domain concentrated only on energy and assumed that performance can be sacrificed as long as it gives better energy. In the low energy domain, performance and energy were given equal importance, and hence, energy delay
product (EDP) was used as our metric. In the medium energy domain, performance becomes even more important and the metric to use must reflect this. Therefore, energy delay-square (ED$^2$) was used for evaluations.

Table 7.1 System Parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Ultra-low</th>
<th>Low</th>
<th>Medium</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Frequency</td>
<td>500 MHz</td>
<td>1 GHz</td>
<td>2 GHz</td>
</tr>
<tr>
<td>L1 Capacity</td>
<td>32 KB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>L1 Latency</td>
<td>1 cycle</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Base L2 Capacity (HP)</td>
<td>128 KB</td>
<td>256 KB</td>
<td>512 KB</td>
</tr>
<tr>
<td>Base L2 Latency (HP)</td>
<td>1.30ns</td>
<td>1.40ns</td>
<td>1.58ns</td>
</tr>
<tr>
<td>Memory Latency</td>
<td>55 ns</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Applications</td>
<td>MiBench</td>
<td>MiBench, SPEC</td>
<td>SPEC</td>
</tr>
</tbody>
</table>

Table 7.2 Evaluated Cache configuration. Capacity is normalized to HP CMOS SRAM.

<table>
<thead>
<tr>
<th>Cache Type</th>
<th>Symbol</th>
<th>Parameters</th>
<th>Capacity</th>
</tr>
</thead>
<tbody>
<tr>
<td>HP SRAM</td>
<td>HP</td>
<td></td>
<td>1.00</td>
</tr>
<tr>
<td>Voltage Scaled</td>
<td>HP-VS</td>
<td>0.5-0.7V</td>
<td>1.00</td>
</tr>
<tr>
<td>LSTP SRAM</td>
<td>LSTP</td>
<td></td>
<td>1.00</td>
</tr>
<tr>
<td>TFET SRAM</td>
<td>TFET</td>
<td>0.3-0.6V</td>
<td>0.64</td>
</tr>
<tr>
<td>Area Opt. STTRAM</td>
<td>STT-Area</td>
<td>-</td>
<td>4.00</td>
</tr>
<tr>
<td>Lat. Opt. STTRAM</td>
<td>STT-Lat</td>
<td>-</td>
<td>2.00</td>
</tr>
<tr>
<td>Power Gated</td>
<td>HP-PG</td>
<td>100,100K,10M</td>
<td>1.00</td>
</tr>
<tr>
<td>Drowsy</td>
<td>HP-D</td>
<td>100,100K,10M</td>
<td>1.00</td>
</tr>
</tbody>
</table>

7.3 Results

7.3.1 Single Core Results

Ultra-Low Energy. In the ultra-low energy domain, the core executes at a relatively low frequency (500MHz) and the capacity of the L2 cache is kept small (128KB) to minimize leakage and dynamic energy. The left part of Table 7.3 shows the energy savings obtained in this domain, which indicates that up to 20% energy savings can be obtained using LSTP, TFET,
and HP-PG configurations. These results show that, TFETs at extreme low voltages consistently provide large energy savings.

**Low Energy.** In the low-energy domain, the frequency of the core and the capacity of the L2 cache are increased (1GHz and 256KB). Further a metric, such as EDP that also considers overall performance is used. Average EDP values for this domain are given in Table 7.3, which show that STT-RAM, LSTP, and TFET caches have the best EDP. While for MiBench applications, LSTP and TFET-0.25V have the best EDP, for SPEC applications that put more pressure on the L2 cache, STT-RAM starts to appear as the best choice in 4 of 10 benchmarks.

**Medium Energy.** Since using larger caches leads to higher cache access latencies, and higher core clock frequencies make caches appear relatively even slower. Table 7.3 shows average ED² values for all cache types and indicates that only HP-D is able to achieve a better score the baseline on average. Energy benefits with other schemes do not appear to be high enough when compared to the performance degradation suffered.

<table>
<thead>
<tr>
<th>Type</th>
<th>Ultra-Low</th>
<th>Low</th>
<th>Medium</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>E_{tot}</td>
<td>Perf</td>
<td>E_{tot}</td>
</tr>
<tr>
<td>HP</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>HP-VS</td>
<td>0.96</td>
<td>1.00</td>
<td>0.98</td>
</tr>
<tr>
<td>LSTP</td>
<td>0.80</td>
<td>0.99</td>
<td>0.89</td>
</tr>
<tr>
<td>TFET</td>
<td>0.80</td>
<td>0.99</td>
<td>0.89</td>
</tr>
<tr>
<td>STT</td>
<td>0.85</td>
<td>0.98</td>
<td>0.91</td>
</tr>
<tr>
<td>HP-PG</td>
<td>0.81</td>
<td>0.83</td>
<td>0.89</td>
</tr>
<tr>
<td>HP-D</td>
<td>0.86</td>
<td>0.98</td>
<td>0.93</td>
</tr>
</tbody>
</table>
Table 7.4 The best cache types and the corresponding evaluation normalized metrics for (top) MiBench and (bottom) SPEC benchmarks.

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>Ultra-Low Energy</th>
<th>Low Energy</th>
<th>Medium Energy</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Ultra-Low Energy</td>
<td>Low Energy</td>
<td>Medium Energy</td>
</tr>
<tr>
<td></td>
<td>Best Config.</td>
<td>$E_{tot}$</td>
<td>Best Config.</td>
</tr>
<tr>
<td>adpcm</td>
<td>TFET</td>
<td>79%</td>
<td>LSTP</td>
</tr>
<tr>
<td>basicmath</td>
<td>TFET</td>
<td>79%</td>
<td>LSTP</td>
</tr>
<tr>
<td>bitcount</td>
<td>TFET</td>
<td>80%</td>
<td>LSTP</td>
</tr>
<tr>
<td>blowfish</td>
<td>TFET</td>
<td>79%</td>
<td>LSTP</td>
</tr>
<tr>
<td>fft</td>
<td>TFET</td>
<td>80%</td>
<td>TFET</td>
</tr>
<tr>
<td>gsm</td>
<td>TFET</td>
<td>81%</td>
<td>TFET</td>
</tr>
<tr>
<td>lame</td>
<td>TFET</td>
<td>80%</td>
<td>TFET</td>
</tr>
<tr>
<td>patricia</td>
<td>TFET</td>
<td>80%</td>
<td>TFET</td>
</tr>
<tr>
<td>qsort</td>
<td>TFET</td>
<td>80%</td>
<td>TFET</td>
</tr>
<tr>
<td>stringsearch</td>
<td>TFET</td>
<td>80%</td>
<td>TFET</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>Low Energy</th>
<th>Medium Energy</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Low Energy</td>
<td>Medium Energy</td>
</tr>
<tr>
<td></td>
<td>Best Config.</td>
<td>EDP</td>
</tr>
<tr>
<td>cactusADM</td>
<td>LSTP</td>
<td>84%</td>
</tr>
<tr>
<td>calculix</td>
<td>STT-area</td>
<td>91%</td>
</tr>
<tr>
<td>dealII</td>
<td>LSTP</td>
<td>90%</td>
</tr>
<tr>
<td>gamess</td>
<td>STT-Lat</td>
<td>74%</td>
</tr>
<tr>
<td>lbm</td>
<td>LSTP</td>
<td>85%</td>
</tr>
<tr>
<td>mcf</td>
<td>STT-Lat</td>
<td>84%</td>
</tr>
<tr>
<td>milc</td>
<td>LSTP</td>
<td>87%</td>
</tr>
<tr>
<td>namd</td>
<td>LSTP</td>
<td>90%</td>
</tr>
<tr>
<td>omnetpp</td>
<td>STT-Area</td>
<td>87%</td>
</tr>
<tr>
<td>sjeng</td>
<td>LSTP</td>
<td>91%</td>
</tr>
</tbody>
</table>

7.3.2 Multicore Results

For the multi-core platform, there is an additional parameter which states that the L2 cache can be private or shared. While a private cache provides isolation of applications, a shared cache can achieve better utilization of the total capacity. In case of private L2 caches, different cache types can be implemented on different cores, which can lead to a heterogeneous private L2 cache. Further, under iso-area constraint, the total cache capacity can also be distributed unevenly across the private L2 caches. Such an uneven distribution can enable capacity sensitive applications to receive higher capacity and operate at a lower miss rate, while resulting in a marginal degradation of the less capacity-sensitive co-runner application.
When working with shared caches, all possible caches presented in Section 7.1 in a homogeneous setting were evaluated, i.e., bank-level or way-level heterogeneity were not considered. However, in case of private caches, the best private cache need not have a symmetric area distribution across the individual caches. Dual-core experiments were run in the low-energy domain using 10 randomly selected workload combinations from the SPEC2006 suite. Figure 7.1 shows the variation in EDP for one of our workloads \( \text{omnetpp-lbm} \) as the area distribution across two private caches is varied. For this workload, optimal EDP is obtained at the ratio of 5:3. To take this into account, asymmetric distributions of area across the two private caches of our processor were also analyzed. In addition, this also included the possibility of using different technologies for each private cache.

Figure 7.2 shows the EDP values of the best private and the best shared cache configurations for all workloads. Note that, in a multiprogrammed setting, EDP is redefined to be the product of total energy and weighted speedup. In workloads consisting of capacity sensitive and insensitive benchmarks such as \( \text{omnetpp-mcf} \), the best private cache is seen to have a better EDP than the best shared cache. This is because the best private cache has a heterogeneous structure.
where each application is free to operate on its own best cache implementation. For instance, the capacity sensitive benchmark (omnetpp) uses an STT-RAM cache to improve its hit rate while the capacity insensitive benchmark (mcf) uses an LSTP cache to reduce its energy. On the other hand, for some other workloads such as calculix-namd, the shared cache configuration is preferred due to better dynamic utilization of the total cache capacity.

7.3.3 Observed Tradeoffs with Emerging Technologies

**STT-RAM Behavior.** When the 4x density and zero cell leakage advantages of STT-RAM over SRAM are considered, one can expect STT-RAM caches to consistently achieve very good results. However, not all applications actually benefit from the increased L2 cache capacity of STT-RAM. Figure 7.3 which compares the L2 cache miss rate of an HP CMOS cache with that of STT-Area and TFET caches. Only a few benchmarks actually show a significant reduction in miss rates. This is because, while read-intensive applications benefit more from the capacity increase of STT-RAM, the long write latency offsets this capacity benefit in case of write-intensive applications [74]. It should be noted that a lower cache miss rate need not
Fig. 7.3 L2 cache miss rates of HP, STT-Area, and TFET (SPEC on low energy domain).

imply a better memory system performance - one must also consider the increased access latency (due to long STT-RAM write pulse) and analyze average memory access time (AMAT). Figures 7.4 and 7.5 show the AMAT and the IPC obtained with STT-RAM normalized to HP baseline. Among the four benchmarks with reduced miss rates, *dealII* has an AMAT value worse than the baseline and only *gamess* shows a significant IPC improvement (18%).

These results indicate that the high write latency associated with STT-RAM restricts the cases where it brings significant performance benefits. Further, due to the leakage of its peripheral circuitry, STT-RAM does not eliminate cache leakage completely.

**TFET Behavior.** Due to its 10T cell structure, TFET caches have 33% less capacity than iso-area HP caches. Therefore, the miss rates with TFET caches, given in Figure 7.3, are higher than those with HP caches. Further, as TFETs are operated at very low voltages (0.25V for TFET vs. 0.8V for HP), they also suffer from higher access latencies. This manifests itself as even higher degradations in AMAT as shown in Figure 7.4. Figure 7.5 indicates that, the overall impact on system performance is an average IPC degradation of 7%.
However, while TFET caches suffer high performance penalties, they are also very energy efficient. Due to the orders of magnitude reduction in off-current, leakage energy is almost entirely eliminated, which gives very significant energy improvements. Analyzing the data in Table 7.3 for TFETs in the low-energy domain shows that, for a 5% reduction in performance, an energy improvement of 11% is obtained. On average, this leads to a significant 7% EDP improvement. Further, in the ultra-low energy domain, the performance degradation reduces to less than 1% and the energy savings reach as high as 20%. Based on these results, one can conclude that when energy is a critical player, using TFET caches can lead to significant energy savings with small performance penalties.

In conclusion, although various circuit-level and technology-based methods to reduce cache leakage energy have been proposed in the literature, from a system designer’s perspective, it is not clear which method shall be expected to be the most effective under a target architectural configuration and workload. In this work, the effectiveness of various leakage reduction techniques for various energy domains and workloads were evaluated. Results indicate that in a strictly energy constrained domain, TFET caches achieve the lowest energy levels. On the other
Fig. 7.5 IPC of STT-Area and TFET normalized to HP (SPEC on low energy domain).

hand, in a relaxed energy constraint scenario, while capacity sensitive applications prefer larger capacity STT-RAM caches, latency sensitive applications prefer lower latency drowsy caches.
Chapter 8

Related Work

This chapter highlights the various prior works in different topics covered in this dissertation. These include works dealing with emerging device technologies, circuit and architectural techniques for optimizing energy, power and temperature in processors, heterogeneous architecture designs and the consequent scheduling of applications using static and dynamic techniques, as well as new innovations like 3D-stacked architectures.

8.1 Emerging technologies for processor architectures

While innovations such as multigate transistor technology (Si FinFETs) have improved the near-threshold characteristics of CMOS to some extent, the physical device limitations are still prevalent at a subthreshold level. These days, many researchers are working on steep slope switches to overcome the thermal energy limitation imposed $60mV/dec$ subthreshold slope. Recently proposed Nano-Electro-Mechanical Switches (NEMS) or NEM relay devices [40] utilize movable electrodes and the instability points between electrical and mechanical forces to achieve this. Other alternatives include reducing the body factor in gate capacitance by using negative-capacitance FET (NC-FET) or exploiting the non-linear energy dependent polarization in ferro-electric gates [88]. The Tunnel FET (TFET) technology described in detail in this dissertation was initially examined in [92]. Saripalli et al. have carried out an architecture-level study of TFET-based designs in [90]. Here, TFET devices are used to design processor cores that have
very high efficiency at low frequencies. There have also been works that examine the reliability of these TFET devices at a circuit and system level, in terms of its soft-error vulnerability [63]. This work establishes that TFETs demonstrate superior resilience to existing CMOS technology at low operating voltages.

8.2 Energy-Efficient Heterogeneous Architectures.

Various efforts in architecture have attacked the dark silicon problem by improving the power consumption of multicores through a series of architectural innovations. Heterogeneous Asymmetric Chip Multi Processors cores have been proposed in the past [56, 57] which deal with the varying demands from the applications in terms of Instruction Level Parallelism (ILP) and Thread Level Parallelism (TLP). These schemes decide the number of big (out-of-order) and small (in-order) cores statically. To circumvent this static techniques, there have been works [47], [42] which dynamically transform the issue widths to cater to different parts of the applications than can exploit the inherent ILP and TLP of the application. [42] employs dynamically fusing the cores to form a large one core group to increase the performance of the sequential code. These fusible cores can also execute parallel threads independently in isolation to speed up the parallel portions of the application. This technique suffers from the overheads while reconfiguring like instruction cache flushes, data migration etc. which are circumvented in [47] by dynamically transforming an Out of order core in to SMT based in-order core. Ipek et al. [41] propose a dynamically reconfigurable processor that can adapt itself to changing work-load behavior dynamically. Showing that critical sections in applications can also be causes for serialization, Suleman et al. [97] illustrate use of a high performance core for critical section execution and low power worker cores for parallel execution. An analysis of heterogeneous
computing using on-chip custom logic, reconfigurable logic, and GPGPU cores is performed in [17]. Venkatesh et al. [104] use processors with specialized datapath cores (or conservation cores), that demonstrate high energy efficiency which mitigates power constraints in a dark silicon environment. The work by Morad et al. [78] shows that heterogeneous multicores can bring significant power reduction for similar performance or significant performance improvement for the same power budget. Various prior works [39, 101, 45] focused on process variations, which is an unintentional source of heterogeneity in multicores. Heterogeneous integration of CMOS with several emerging devices has been an active research area due to the shortcomings being exposed in conventional CMOS with technology scaling, which is one of the primary aspects that I focus on in this thesis.

8.3 Application Mapping on Heterogeneous Architectures

Various efforts have been made to attack the problem of how to map applications to multicore processors. In [7], the behavior of commercial applications running on asymmetric multicores is analyzed. They indicate that the application itself needs to be aware of asymmetry, and stress that when running multithreaded OpenMP applications on heterogeneous systems, guided scheduling typically performs better. Bhaduria et al. [8] analyze multithreaded applications running on a frequency-heterogeneous real multicore system. They point out the use of hardware counters in quantifying application progress, which can in turn be used to determine the best number of threads to use at runtime. Luk et al. [69] also point out the difficulty in mapping applications to cores on a heterogeneous processor and indicate that this is no longer a task that can easily be performed by the programmer. To simplify the process, they propose an
automated method that performs runtime adaptive mapping of computations to processing elements. These prior works agree on the fact that simple static work partitioning constructs do not perform well for heterogeneous processors and some form of dynamic load balancing is required to better make use of the performance asymmetry across cores. None of the prior works consider accompanying dynamic work partitioning across application threads with dynamic power management across cores as a method to better utilize heterogeneous multicore processors, which is also a contribution of this dissertation.

8.4 3D stacked architectures

Incorporation of 3D stacking technology into chip design has gained popularity due to the significant benefits that it offers, such as smaller package size due to reduced area footprint, decrease in communication power and delay on account of shorter wires, and increased bandwidth due to the higher availability of pins. While the viability of memory-on-memory and memory-on-core stacking has been demonstrated [66, 35], thermal and yield limitations continue to be of concern, especially in the case of core-on-core stacking, as has been discussed in [67]. There have been several works that discuss the feasibility of 3D integration on chip. For example, cost-effective techniques for successful 3D integration has been examined in [14]. The authors in [29] analyze various aspects of heterogeneous device integration in 3D technology. Our work, while based on this concept, extends this further by incorporating thermal and yield models as well as application mapping algorithms on heterogeneous 3D architectures. [36] proposes the idea of Resource Pooling across multiple stacked layers within a processor and efficiently allocating resources and tasks to each layer. Our work examines the viability of stacking cores on top of
each other as opposed to folding individual cores across multiple layers, which is an extremely complicated task due to wiring concerns.

8.5 Architectural techniques for power and thermal aware execution

Dreslinski, et. al [26] proposed the concept of Near Threshold Computing (NTC), where the processors are run at a voltage level close to \( V_t \), so that it operates at a point where the power efficiency is the highest. By incorporating TFET cores into our system, the near-threshold voltage restriction is eliminated and a far wider range of operating points are available to the user. Computational Sprinting is another technique which can be used to improve the utilization of processors under power and thermal limitations [86]. In this paper, the authors briefly allow the processor to briefly exceed the processor power limitation by operating it at extremely high performance points for short periods of time. Our system allows for high performance CMOS cores to selectively run based on the application requirement, which covers most of the scenarios for which sprinting is required. It is also a more robust technique from the perspective of reliability and aging.

8.6 Thermal-Aware application mapping on multicores

In [94], the authors propose PROMETHEUS, a heterogeneous multiprocessor SOC-based thermal-aware scheduling policies, such as TempoMP and TemPrompt. [21] proposes a proactive thermal management technique in MP-SoCs. Unlike these methods that operate over a specific domain, the techniques proposed in this thesis can be extended over a wide range of application domains and incorporated everywhere from embedded SOCs to high end server
architectures. [34] describes a thermal-aware DVFS algorithm for real-time applications running on a multicore, while [15] proposes DVFS techniques on 3D multicores.

8.7 Architecture and technology innovations in last-level cache design

When sources of power consumption in processors are investigated, large last-level caches appear to make an important contribution [51, 49]. Reducing cache leakage has been the subject of significant amount of research in recent years [37, 79, 31, 106]. Circuit-level techniques such as power gating, drowsy caches, and voltage scaling have been proposed to reduce cache leakage. Alternatively, technology replacement oriented studies proposed emerging devices such as Spin-Transfer Torque RAMs (STT-RAMs) [37, 46] and Tunnel-FETs [77] to replace CMOS SRAM with similar goals. Chapter 7 of this dissertation carries out a joint study of both technology and architecture-based techniques for energy-efficient LLC designs.
Chapter 9

Conclusion

With scaling of transistor gate lengths to orders of a few nanometers, physical constraints such as subthreshold-slope limitations and short-channel effects are beginning to limit the performance and power benefits expected from each new generation. Innovations that are purely restricted to the architecture and micro-architecture levels of abstractions are proving to be insufficient in overcoming these hurdles. In such a scenario, it becomes mandatory to explore new device technologies that possess characteristics that are superior to the current technology. However, there are certain tradeoffs in terms of performance, energy and reliability that these devices entail, with respect to the state-of-the-art CMOS transistors. Hence it becomes necessary to examine techniques for device-architecture co-design in order to gauge the system impact of these new technologies.

In my dissertation, I have examined various possible avenues for computing in the post-CMOS era. I have studied several heterogeneous innovations for optimizing the power and performance of both the core and the memory hierarchy in a processor. These include examining a whole range of emerging technologies that could potentially complement or even, replace current CMOS technology within the next few technology generations. Among these new devices, steep slope transistors like the Heterojunction Tunnel Field Effect Transistors (HTFETs) have been shown to demonstrate such favorable characteristics. By proposing a device-architecture abstraction model, I have attempted to study the impact of these emerging device characteristics
at the micro-architecture and architecture levels of abstraction. I have studied the tradeoffs between device and architecture modifications for a whole range of application domains by investigating CMOS-TFET heterogenous multicore designs and proposed various static and dynamic techniques to optimally map workloads to such device-heterogeneous multicores under power and temperature constraints. I have extended the study to encompass non-traditional architecture techniques such as 3D stacking of processors and domain-specific accelerator designs. Finally, I have also explored the impact of such a device-architecture co-design on the memory hierarchy, particularly on on-chip last-level caches.

9.1 List of publications:

- **VLSI 2015**: *Thermal-aware application scheduling on device-heterogeneous embedded architectures*,
  
  Karthik Swaminathan, Jagadish Kotra, Huichu Liu, Jack Sampson, Mahmut Kandemir and Vijaykrishnan Narayanan, VLSI Design Conference (VLSI), Bangalore, India, Jan 2015

- **SIPS 2015**: *Understanding the Landscape of Accelerators for Vision*,
  
  Nandhini Chandramoorthy, Karthik Swaminathan, Matthew Cotter, Xueqing Li, Indranil Palit, Kevin Irick, Sharon Hu, Michael Niemier and Vijaykrishnan Narayanan, Signal and Image Processing Symposium(SIPS), Belfast, UK 2015

  
  Siddharth Advani, Nandhini Chandramoorthy, Karthik Swaminathan, Kevin Irick, Yong
Cheol Peter Cho, Jack Sampson and Vijaykrishnan Narayanan, International Conference on Computer Design (ICCD), Seoul, South Korea, Oct, 2014

- **CSICS 2014**: *Enabling Power-Efficient Designs with III-V Tunnel FETs*,
  Moon Seok Kim, Huichu Liu, **Karthik Swaminathan**, Xueqing Li, Suman Datta and Vijaykrishnan Narayanan, Compound Semiconductor IC Symposium (CSICS), San Diego, CA, Oct, 2014

- **ISCA 2014**: *An Examination of the Architecture and System-level Tradeoffs of Employing Steep Slope Devices in 3D CMPs*,

  **Karthik Swaminathan**, Huichu Liu, Xueqing Li, Moon Seok Kim, Jack Sampson and Vijaykrishnan Narayanan, Design Automation Conference (DAC), San Francisco, CA, June, 2014

- **DATE 2014**: *Modeling Steep Slope Devices: From Circuits to Architectures*,
  **Karthik Swaminathan**, Moon Seok Kim, Nandhini Chandramoorthy, Behnam Sedighi, Robert Perricone, Jack Sampson and Vijaykrishnan Narayanan, Design Automation and Testing in Europe (DATE), Dresden, Germany, Mar, 2014

- **IEEE Micro 2013**: *Steep Slope Devices: From Dark to Dim Silicon*,
• **CODES+ISSS2012**: *Performance Enhancement under Power Constraints using Heterogeneous CMOS-TFET Multicores*,


• **ISLPED 2012**: *Design Space Evaluation of Workload-specific Last Level caches*,


• **ASPDAC 2012**: *When to forget: A system level perspective of non volatile memories*,

**Karthik Swaminathan**, Raghav Pisolkar, Cong Xu, Vijaykrishnan Narayanan, Asia-South Pacific Design Automation Conference (ASPDAC), Sydney, Australia, Jan 2012

• **ISLPED 2011**: *Improving Energy Efficiency of Multi-Threaded Applications using Heterogeneous CMOS-TFET Multicores*,


• **ISVLSI 2011**: *Towards Resilient Micro-Architectures: Datapath Reliability Enhancement using STT-RAM*,

**Karthik Swaminathan**, Ravindhiran Mukundrajan, Niranjan Soundararajan and Vijaykrishnan Narayanan, International Symposium of VLSI Design (ISVLSI), June 2011, Chennai, India
• GLSVLSI 2011: Enabling Architecture Innovation Using Non-Volatile Memory,
Vijaykrishnan Narayanan, Vinay Saripalli, Karthik Swaminathan, Ravindhiran Mukundrajan, Guangyu Sun, Yuan Xie and Suman Datta Great Lakes Symposium of VLSI Design, 2011, Lausanne, Switzerland
Bibliography


[32] R. Gandhi et al. Vertical Si-Nanowire n-Type Tunneling FETs With Low Subthreshold Swing (≤ 50mV/decade ) at Room Temperature. IEDM, 2011.


[76] D.K. Mohata et al. Self-aligned Gate NanoPillar In0.53Ga0.47As Vertical Tunnel Transistor. In DRC, 2011.

[77] S. Mookerjea et al. Experimental Demonstration of 100nm Channel Length In0.53Ga0.47As-based Vertical Inter-band Tunnel Field Effect Transistors (TFETs) for Ultra Low-Power Logic and SRAM Applications. In IEDM, 2009.


Karthik Swaminathan is a Ph.D student in the Department of Computer Science and Engineering at the Pennsylvania State University. He joined the department in Fall 2009 as part of the Microsystems Design Laboratory under Prof. Vijaykrishnan Narayanan. His research is primarily focused on power aware computer architectures and on leveraging emerging device technologies in the architectural domain to improve performance, power and reliability. He graduated from the Indian Institute of Technology, Madras in 2008 with a Dual Degree (Bachelors and Masters) in Electrical Engineering. Prior to beginning his PhD, he worked for a year at IBM India Research Labs, New Delhi as part of the High Performance Computing Group. He was awarded the prestigious IBM Ph.D Fellowship for the year 2012-2013. He worked as a co-op intern at IBM SRDC, East Fishkill and IBM TJ Watson Research Center, Yorktown from July to December 2012 with the 3D embedded DRAM design group at IBM SRDC headed by Toshiaki Kirihata and the Power-aware Architecture group headed by Pradip Bose at IBM Research. In the summer of 2013, he interned with the Components Research Group at Intel with the Exploratory Integrated Circuits Group headed by Ian Young, primarily focusing on evaluating the energy-performance tradeoffs of Tunnel-FET based architectures.