VALUE-AWARE MULTI-OBJECTIVE INTERCONNECT OPTIMIZATION

By

Sharath Jayaprakash

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

Electrical Engineering

2011

ABSTRACT

VALUE-AWARE MULTI-OBJECTIVE INTERCONNECT OPTIMIZATION

By

Sharath Jayaprakash

Geometrical scaling of critical semiconductor dimensions while keeping electric field constant improves logic delay and power consumption, but degrades global and semi-global interconnect delay. Repeater insertion and higher aspect-ratio wires partially alleviate interconnect delay, but adversely impact power consumption, temperature, and crosstalk noise. Although a number of interconnect design techniques have been proposed, they primarily target: (1) one or the other design metric, not all relevant metrics simultaneously and/or (2) random or worst-case traffic conditions instead of real-workload traffic, which is important because interconnect behavior (energy, delay, temperature, etc.) depends upon traffic value characteristics.

To address the shortcomings of previous work, we present a value-aware interconnect design methodology that permits simultaneous optimization of multiple metrics in a prioritized manner tailored to target application requirements. Towards this end, we first survey existing commonly-used interconnect design techniques and analyze their influence on different interconnect circuit parameters and design metrics. From among these, we focus on wire spacing and data encoding as examples of two techniques to which we apply our value-aware methodology for optimization of interconnect energy, average delay, and peak wire temperature in the context of the SPEC CPU2k benchmarks. Next, we describe and compare two value-aware techniques that optimize energy savings within area constraints: partitioned hybrid encoding and value-aware wire spacing. Finally, we present three encoding techniques with increasing flexibility to adapt to traffic value characteristics: (1) *static*, which employs a single encoding mode, fixed at design time based on traffic value characteristics, in all

cycles; (2) *dynamic*, which chooses the most suitable encoding mode in any given cycle from among two or more predetermined modes; and (3) *adaptive*, which selects the most appropriate encoding mode in any given cycle from among a set of modes that changes with traffic conditions at run time. These techniques provide increasingly greater benefits, with our static technique easily outperforming previous, more complex, but value-oblivious, dynamic encoding techniques.

Dedicated to my Parents, Jayaprakash and Lakshmi Banu,

Leela Avva, Sulochana Avva, and Suri Mama

for their unconditional love and support

ACKNOWLEDGEMENTS

The completion of my doctoral research has been the most significant academic challenges that I have ever undertaken. This endeavor would not have been possible without the support and guidance of many people. I wish to express my appreciation to all of them.

First and foremost, I would like to thank my advisor, Prof. Nihar R. Mahapatra, for his patience, support, and guidance during the course of my research. Under his tutelage I have developed abilities to identify, analyze, and present research problems and solutions in a systematic and articulate manner. I would also like to thank my dissertation committee members, Prof. Richard Enbody, Prof. Micheal Shanblatt, and Prof. Pang-Ning Tan for their insightful review and comments which have helped me improve this work.

I also greatly appreciate the staff at MSU, especially Fred Hall (DECS) and Andrew Keen (HPCC), for their technical support.

I would like to express my gratitude to my cousin Nandini for her encouragement and motivation to pursue doctoral research. I am indebted to all the excellent teachers I have had over the years, particularly Mr. Ramachandran for his interest in my early education.

I have also been fortunate to be in the company of a lot of good friends, from grade school to graduate school, and I thank them all for their support. I have cherished the friendship of my lab-mates Krishnan Sundaresan and Kaushal Gandhi and I am grateful for their invaluable guidance during the early stages of my research. I would also like to thank my friend Keyur Desai for his support, wisdom, and intellectual feedback.

I would like to remember my loving brother Bharath whose memories have helped me keep perspective in life and deal with reality.

This dissertation is dedicated to my mom, Lakshmi Banu, dad, Jayaprakash, Leela avva, Sulochana avva, and Suri mama for their unconditional love and support which has carried me to where I am today.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

The Need for Value-Aware Interconnect Design

Technology scaling imposes limitations on interconnect design for high-speed and low-power digital circuits and systems. Reducing transistor physical dimensions increases circuit density and speed, and reduces energy consumed per logic operation. However, microprocessor performance increase is only roughly proportional to square root of increase in logic complexity (Pollack's Rule) due to increase in global interconnect delay that reduces or cancels the speed gained due to smaller transistors [1, 2]. Reduced interconnect dimension increases parasitic resistance, inductance, and capacitance that aggravate interconnect delay, power consumption, and signal reliability. Thus, design of on-chip interconnects is one of the most important challenges in nanometer-scale ICs [3].

1.1 Interconnect Problems and Challenges

Global interconnect delay does not scale with gate delay as increased integration leads to longer and narrower wires. This problem in global interconnect scaling is evident from the fact that in the older 1.0 μm Al/SiO_2 technology transistor delay was $\sim$20 ps and interconnect RC delay for a 1 mm was $\sim$1 ps while in the projected 35 nm Cu/low-k technology

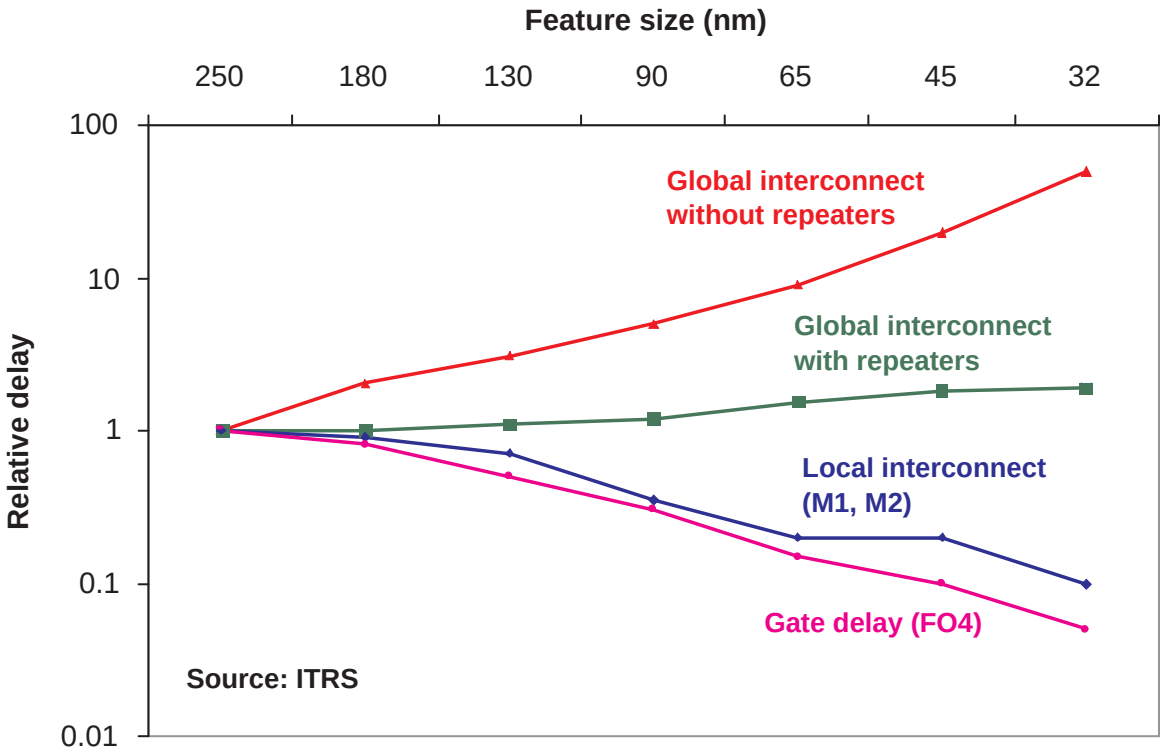


Figure 1.1: Local interconnects scale with gate delay whereas global interconnect delays do not [4]. "For interpretation of the references to color in this and all other figures, the reader is referred to the electronic version of this dissertation."

transistor delay will be ∼1 ps and interconnect RC delay will be ∼250 ps [1, 5]. Further, Cu resistivity increases in sub-100 nm technologies, as shown in Figure 1.2, as wire width becomes comparable to the mean free path of charge carriers and aggravates interconnect delay and resistive noise (IR drop). This limitation of interconnect delay affects latency and bandwidth of communication channels and has a significant impact on system performance.

Further, in Intel microprocessors, interconnect power accounted for approximately 51% of microprocessor power in the 130 nm technology and was projected to rise up to 65%-70% of microprocessor power, over the next few years [7]. These trends in interconnect power continues to hold even in modern interconnect-centric architectures, such as multi-core, system-on-chip (SOC), and network-on-chip (NOC), where interconnect energy is expected to account for almost 70% of the total chip energy [8].

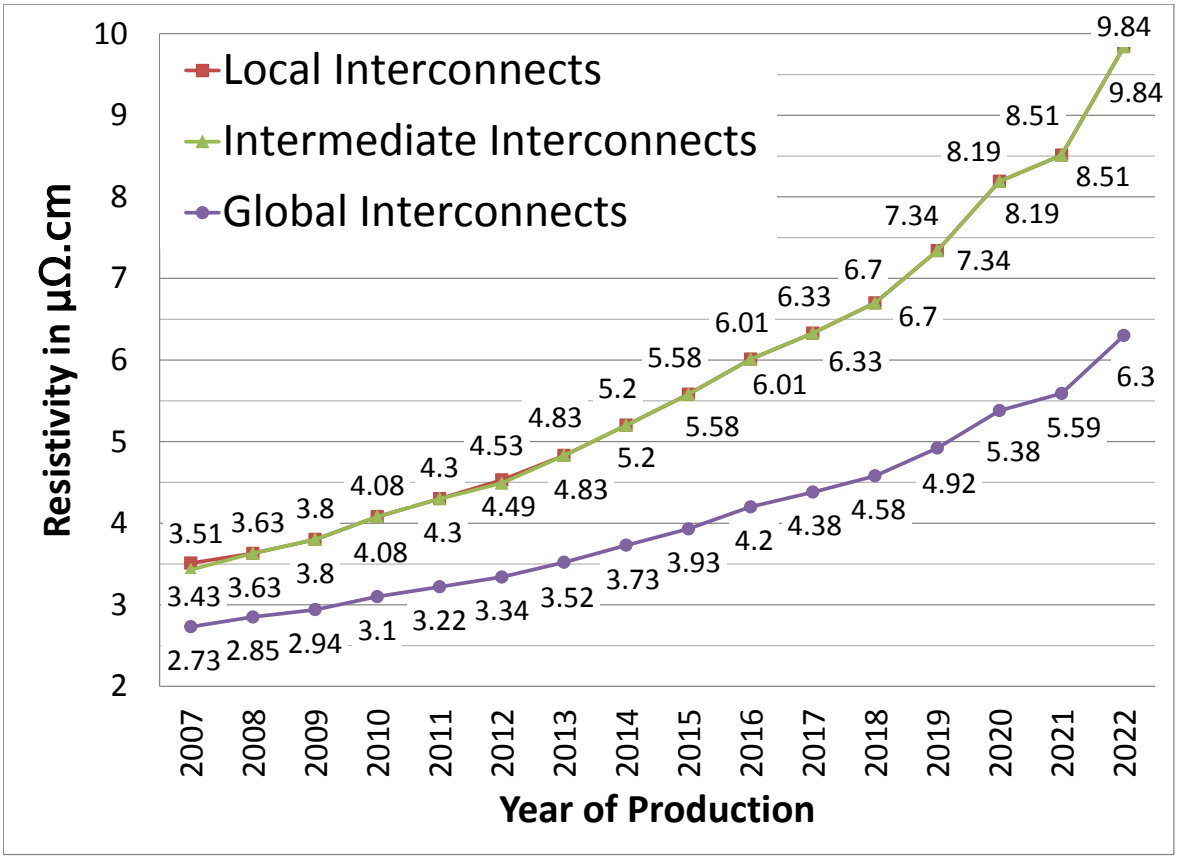


Figure 1.2: Effective copper resistivity ($\mu\Omega$.cm) increase due to both grain boundary and interface electron scattering [1, 6].

This increased energy dissipation leads to higher wire temperature in global interconnects which are furthest from the substrate/heat sink. Figure 1.4 shows the increasing temperature gradient between substrate and top metal layer due to interconnect scaling. Top metal layer, used for routing global interconnects, is expected to reach temperatures as high as 209°C in 45 nm technology, for worst-case scenario where interconnect stack carries currents with maximum rated current density [9, 10]. The temperature gradient for 35-nm technology is smaller than that for the 50-nm technology due to larger fraction of metallization (Cu) layers compared to inter-layer dielectric (ILD) layers, a product of the ITRS scaling scenario used in the analysis. The maximum interconnect temperature also raises the peak RC delay, thereby degrading both performance and reliability. Moreover, meso-porous low-k dielectrics, expected to be introduced in 2012 [1] to helped alleviate the impact of interconnect scaling on

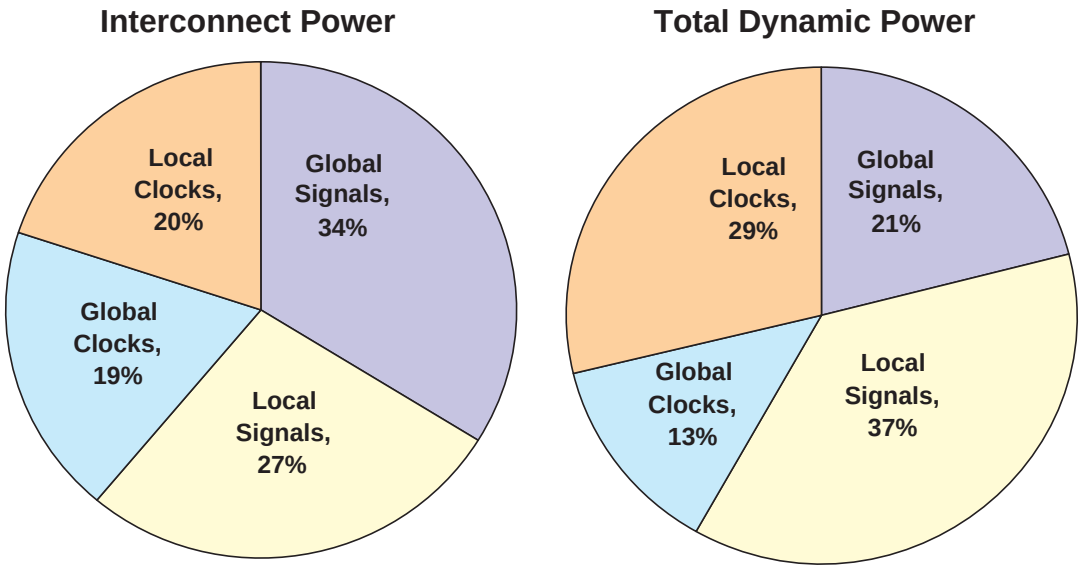


Figure 1.3: Global lines are responsible for 21% of total dynamic power dissipation at 130 nm [7].

performance and energy dissipation, are mechanically weaker and aggravate thermal issues in interconnects, due to increased Joule heating [11], which lead to increased interconnect delay [12, 13, 14] and stress related breakdowns such as electromigration (EM), and time dependent dielectric breakdown (TDDB) [11, 15, 16]. Further, the maximum current density limits for the current dielectric cap technologies for copper will be exceeded by 2013 [1], exacerbating interconnect reliability.

The impact on interconnect performance, power, and temperature created by scaling of conventional metal/di-electric systems cannot be addressed by technology means alone, such as use of low-k dielectric and low resistance Cu metal. Improved layout and circuit design techniques (staggered placement of buffer, sleep transistors, etc.) and novel architectures (multi-core, network-on-chip (NOC), etc.) have been successful in addressing interconnect design limitations. However, unlike new materials like low-k dielectric, newer architectures require new design tools and new software, and are not applicable to all designs. Hence,

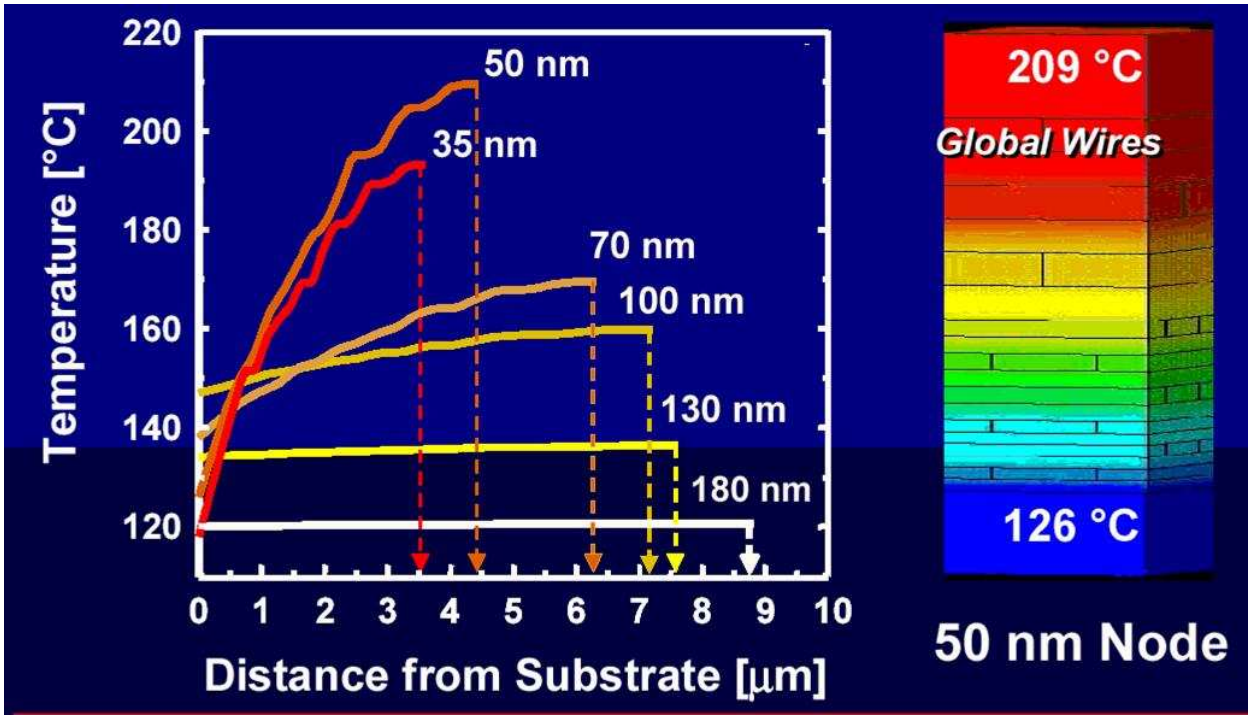


Figure 1.4: Global wire temperatures are expected to reach as much as 209°C in 45 nm technology [9].

interconnect remains a bottleneck for many applications, even with these advances [1].

While RC delay is still a major factor in digital applications, increasing aspect ratio of wires has made capacitive coupling between two neighboring wires one of the most dominant sources of interconnect delay and energy. Figure 1.5 shows increase in capacitive crosstalk, which increases interconnect delay by as much as 300% and also causes delay uncertainties [1]. Moreover, the variability in interconnect energy and delay, due to deviations from ideal interconnect shape, shown in Figure 1.6, caused by process variation, degrades interconnect delay. Further, greater integration, higher frequency, and lower supply voltage, due to scaling, have aggravated interconnect signal integrity issue due to inductive and resistive noise of the power distribution network. Hence, various levels of interconnect simulation need to be coupled with design in a bi-directional manner to reduce design cycle time.

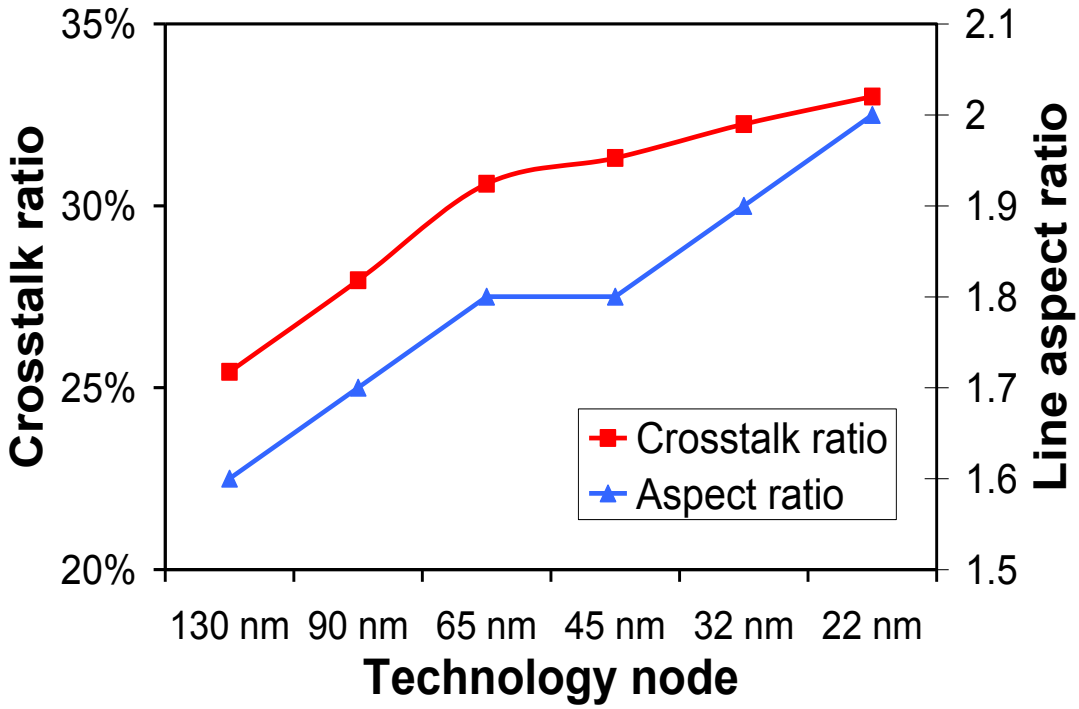


Figure 1.5: Increase in capacitive coupling due to higher aspect ratio [1].

1.2 Potential Interconnect Solutions

Interconnect delay is still the most important factor in digital applications. However, technology scaling and higher operating frequency increase the importance of capacitive and inductive coupling effect which dominates interconnect delay, energy, and noise. In addition, process variations leading to deviations from ideal interconnect shapes influence interconnect design. To solve these problems many options have been proposed, including alternatives to the metal/dielectric system. Solutions such as Cu/low-k dielectric, three-dimensional interconnect structure, short wire architecture, network on-chip (NOC) will help alleviate interconnect scaling in current and future technologies. The pros and cons of these solutions are studied next.

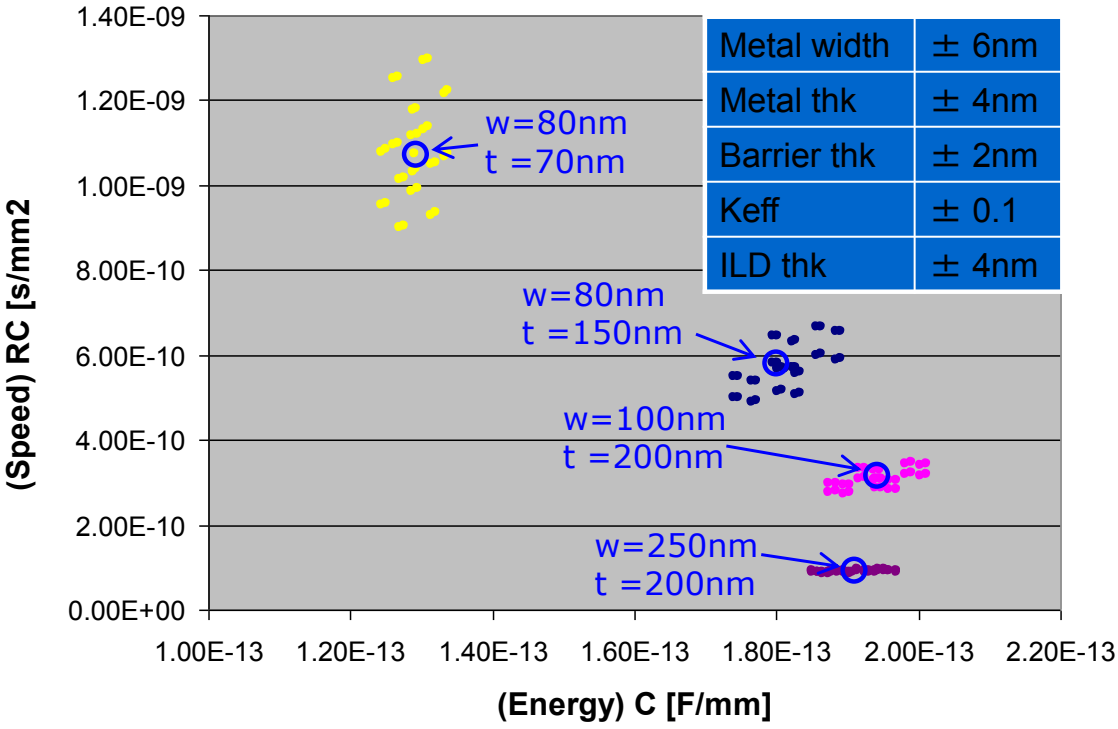


Figure 1.6: Variability in interconnect delay and energy due to process variation. [1].

1.2.1 Newer Materials and Process Enhancements

IBM introduced copper interconnect in its high speed microprocessor; 400 MHz PowerPC 750, to reduce interconnect resistance. Although resistivity of cooper is 40% less than that of aluminum, the percentage of performance improvement was limited to 15%, for 0.25 μm technology, due to gate delay. However, performance benefits will increase as interconnect delay dominates critical path. For similar reasons, use of low-k dielectric, using k=2.5, to reduce interconnect capacitive load, yields a performance improvement of only 25% [17] compared to the conventional silicon dioxide, which has k=3.9. However, use of Cu and low-k dielectric reduces interconnect reliability during both fabrication and chip lifetime due to aggravated thermal issues and low mechanical strength. While, use of Cu-Al alloy and modification to copper surface, to form CuSiN, can significantly reduce electromigration and improve reliability these techniques increase interconnect resistance and reduce yield due to

metal shorts or leakage, respectively [18, 1].

Newer materials and process enhancements is probably the only solution that can be applied, universally, to all IC products. However, such advances usually take many years and cannot completely alleviate interconnect challenges due to scaling which typically occurs every 18 months. Hence, solutions based on new signaling methods, innovative design techniques and new architectures are required to extend the conventional metal/dielectric system.

1.2.2 Novel Design Techniques

Circuit Design Techniques

Many circuit designs have been proposed to limit interconnect delay on critical path. Techniques such as buffer/repeater insertion, wire shaping, wire spacing, low-swing differential voltage and current sensing have been proposed to reduce interconnect delay and/or power. However, most of these techniques require additional chip area. Further, repeater insertion increases power consumption while low-swing differential voltage and current sensing are highly susceptible to noise, due to reduced noise margins. In addition, adiabatic techniques based on multiple supply voltage and charge sharing/recycling addresses interconnect power but increases interconnect delay.

New Architectures

Modular architectures, such as systolic arrays, multi-core architecture, and NOC, that reduce the need for fixed length lines have becomes popular and address some of the performance bottleneck due to interconnect. Nevertheless, multi-core architectures used in the design of modern general purpose microprocessor are limited by the lack of parallel programming in applications and hence fail to exploit the performance benefits due to availability of multiple processor cores. In addition, most new architectures are not applicable to all processor designs and are only useful in specific applications, such as pattern recognition, multiprocessor

systems, and arithmetic computation, and are limited by tools and design methodologies completely different from conventional microprocessors.

3D ICs

Stacking of active devices, using 3D interconnects, reduces the line length and signal propagation delay. Such 3D interconnects could potentially reduce global interconnect delay and power by reducing wire length. While potential delay and power advantages depends on a specific 3D implementation, calculations show reductions in global interconnect length by as much as 50%, resulting in 4× reduction in delay and 2× reduction in energy [19]. Since, active devices are stacked the die area is also reduced by almost 50%. It is clear that major improvements are likely possible using 3D interconnect although there are many factors, not accounted for in a simple 3D IC model that could reduce the performance advantages.

While different approaches to 3D IC integration have been proposed, bonding and alignment of two layers of die, wafer thinning, and etching of high density through-silicon vias (HDTSVs) continue to remain a major problem in the fabrication of 3D ICs. Other major challenges include models of manufacturing costs and yield, thermal modeling and heat removal, probing and testing for stacked structure, reliability of 3D stacks, standardization of 3D processes, and CAD tools for optimal design [1].

1.2.3 Beyond Metal/Dielectric Systems

Optical Interconnect

Optical cables have been used in the communication industry for long-line cables. Optical interconnects, proposed to replace metal wires, provide low latency and large bandwidth because of speed-of-light propagation and the ability to accommodate multiple wavelengths on a single optical path. This manifold increase in data carrying capacity provides bandwidth densities not achievable by electrical means. In addition, crosstalk between signals is minimized due to minimal interaction between photons and ability for light to propagate in

waveguides or free space. This results in low skew and jitter clock distribution and very high signal integrity.

The focus of optical interconnect is on cost efficient implementations that take advantage of the unique properties of optical architectures to increase overall system performance. They are not expected to totally replace Cu/low-k systems because of pitch constraints, as well as delay and power considerations. However, the development of CMOS-compatible optical components is of paramount importance for optical solutions to be viable. However, this area is not yet sufficiently mature though significant progress has been made in this direction [1].

Carbon Nanotubes (CNT)

CNT are rolls of graphene sheets, which are one atom thick carbon layers. CNT are considered a potential solution to improve interconnects performance, energy, and reliability because of their higher electrical and thermal conductivity, and resistance to electromigration compared to conventional interconnects. However, CNT integration faces numerous technical challenges. The selective growth of metallic single-walled CNT and high density integration of CNT to achieve desired conductivity, horizontal growth of CNT, developing low resistance contacts, and to grow CNT at lower temperatures with very low defect density are some of the important challenges that need to be addressed to realize CNT integration [1].

1.3 Shortcomings of Existing Approaches

Real-world workloads cause traffic that exhibits significant spatial, temporal, and value locality which impacts interconnect design metrics, such as delay, energy, temperature, and reliability, which are switching activity dependent. However, almost all existing interconnect design techniques are developed without an accurate knowledge of traffic characteristics. Some of the techniques that use average switching activity factor [20, 11] to study interconnect design metrics lead to inaccurate design as traffic characteristics are information

and time dependent. The type of information (data, instruction, or address) being transmitted has a significant impact on switching activity. For example, data traffic is expected to be more random than either instruction or address traffic as data streams are less predictable than either instruction or address. In addition, L1 cache hits causing extended periods of idle cycles on interconnection between L1 and L2 cache can lead to inaccurate energy and temperature estimation which has a significant impact on wire delay and electromigration. Hence, design techniques that ignore information and time dependence of activity lead to inaccurate design. Hence, design techniques tuned to information and time dependent activity-factor are needed for interconnect design.

Existing design technique also ignore the trade-offs between various interconnect design metrics. For example, increasing space between two interconnect reduces the energy dissipation, by reducing inter-wire coupling capacitance, however they could potentially increase interconnect wire temperature by increasing thermal resistivity (dielectric between wires is a poor conductors of heat). Such trade-offs, if not analyzed, leads to more iterations between high-level design and physical layout which results in increased design time and time to market. Therefore, optimization framework used in making early-stage design decisions is required to achieve faster design closure.

1.4 Our Approach and Contributions

Geometrical scaling of CMOS has provided improved cost per function, speed, power, and reliability to satisfy Moore's law over the last few decades. However, due to physical limitation of CMOS and increased cost of newer technologies, multiple new devices, and new manufacturing and design paradigms are required to work in conjunction with geometric scaling to provide equivalent scaling, to satisfy Moore's law. In this dissertation we provide novel interconnect design approach, that exploits the data dependent nature of interconnect design metric and program characteristics of real-world applications such as SPEC benchmarks.

1.4.1 Value-Aware Interconnect Design

Existing techniques that target interconnect design metrics, such as delay, energy and temperature reductions, perform well only when information is randomly distributed over time. However, information transmitted in microprocessors show high degree of correlation across programs as well as across different phases of a program, due to the presence of temporal, spatial, and value localities. Temporal locality describes the likelihood that a recently referenced item will be referenced again soon, while spatial locality describes the likelihood that a close neighbor of a recently referenced item will be referenced soon. Value locality, on the other hand, refers to the likelihood that a value produced by an instruction is the same as a value produced by another recently executed instruction.

Program characteristics such as temporal and spatial locality (also known as locality of reference) has been exploited to improve microprocessor performance using techniques such as caching, branch prediction, etc., while value locality has been exploited for value prediction [21], design of value centric cache [22] and register files [23], etc. to improve processor performance. In addition, knowledge of circuit operations over a variety of benchmarks has also been critical in the effective implementation of techniques such as clock gating [24], power gating [25] and register sub-banking [26]. However, almost all interconnect design techniques perform well only when values transmitted are randomly distributed in time and do not exploit substantial amounts of temporal, spatial, and value locality in address, instruction, and data traffic in microprocessor.

The reason for locality in various information traffic are:

1. **Address traffic**: Instruction addresses issued by the processor to the L1 cache are typically sequential, except after a branch or jump statement, even then the target addresses are not typically very far away from the last address. This is the reason why branch and jump instructions use PC-relative addressing. Also, data addresses exhibit these localities primarily because data arrays scanned using loops are placed in contiguous memory locations.

2. **Instruction traffic**: Instructions executed by a processor correspond to instruction addresses and hence instructions exhibit the same temporal and spatial locality as instruction addresses. In addition, instruction set limits the possible values that could be transmitted in the stream. Also, not all instructions, opcodes, register operands, and immediate constants occur equally in a program, leading to more predictability behavior of instruction stream.

3. **Data traffic**: Data processed also exhibit temporal and spatial locality, although to a lesser extent than addresses and instruction traffic. Though one might assume data value to be random, the magnitude of values communicated and stored in registers and caches, and/or CAM structures in the processor core are small and often predictable [27]. This concept of value locality adds redundancies in data traffic. Also, not all data types (integer, floating point, character, etc.) are equally likely.

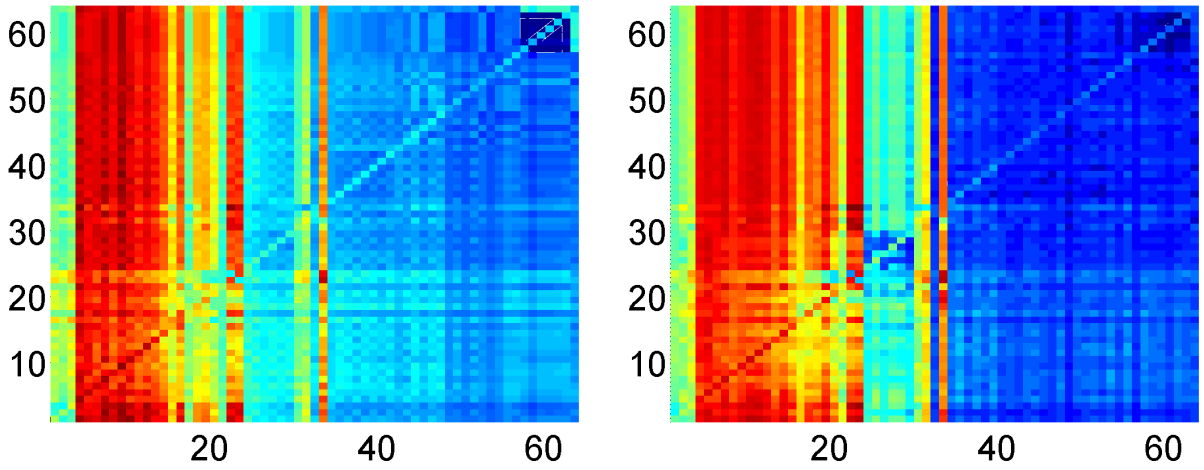


Figure 1.7: The transition activity for each bit on the data bus, including interwire coupling activity for training and test programs, respectively.

Redundancy in traffic, due to temporal, spatial, and value locality, emphasizes that interconnect design should be based on activity characteristics of different types of information transmitted during execution of typical applications. Such design can be achieved with the following steps: (1) Use cycle accurate microarchitecture simulators to execute real-world applications and profile the information transmitted, (2) identify correlation in traffic and

their impact on interconnect design metrics, (3) and design techniques that optimizes the interconnect metrics. To show the high correlation in switching activity, in real-world applications, we plot the transition characteristics of the data bus, between processor and L1 cache, for two sets of programs, selected randomly from SPEC CPU 2000 benchmark. The likeness of transition activity, between the two sets of programs, shown in Figure 1.7, affirms that value-aware interconnect design obtained using a representative workload, is likely to work well for any real application in the same domain due to the similarities in program characteristics.

1.4.2 Novel Average Delay Minimization Technique

The rate at which signals can be transmitted in a high-speed processor bus is decided based on the worst-case crosstalk pattern. This pessimistic estimation gives rise to significant performance penalties since the worst case never occurs or occurs with very low frequency in actual applications. Hence, an adaptive bus design called variable cycle bus (VCB) architecture that examines incoming data patterns and transmits them using variable number of clock cycles, improves bus performance significantly. To maximize effectiveness of our adaptive bus architecture, we develop value-aware encoding techniques that minimize the occurrence worst-case crosstalk scenarios. Results on SPEC CPU 2000 benchmarks, in a workload-specific optimization scenario for static signaling and bit-ordering encoding technique, reduces the number of transmission cycles by 21.24% and 10.90% cycles for data and instruction traffic.

1.4.3 Enabling Early-Stage Design Exploration

The architecture community is limited by the absence of accurate models to predict the performance, power and reliability of a system while the accurate models used by the VLSI community are too complex to be incorporated at design time. One of the objectives of this dissertation is to bridge this gap by building highly accurate interconnect models that

capture delay, energy, temperature and reliability to enable more robust early-stage design exploration. We use these highly accurate models in microarchitectural level simulators to determine the impact of activity-dependent interconnect design metrics, such as delay, energy, and temperature. This enables interconnect optimization earlier in design cycle instead of later, where opportunities for interconnect optimization are limited. Including interconnect design at an early stage in the design cycle allows more flexibility in design exploration and empower designers to optimize architectures for performance, power, and cost.

1.4.4 Multi-Objective Optimization Framework

Most of the interconnect design metrics, such as delay, energy, and temperature, which depend on data activity can be addressed effectively using value-aware interconnect design. However, techniques used to address interconnect delay could potentially impact energy, negatively. Similarly, energy efficient techniques could have a negative impact on temperature, though this statement seems counter intuitive. For example, increasing space between wires and/or reducing wire width reduce interconnect energy dissipation, yet this could increase the wire temperature as dielectrics surrounding the wire are poor conductor of heat. Also, this is true for other energy optimization techniques as peak temperature of the bus depends on energy density rather than the overall energy dissipated. Hence, there is a need for a unified framework that can analyze various design techniques and their impacts on all interconnect design metrics.

1.5 Dissertation Outline

This remainder of this dissertation is organized as follows. Next, in Chapter 2, we presents a survey of interconnect design metrics and techniques, while Chapter 3 provides an overview of data dependent interconnect models, experimental methodology and simulation infra-

structure. Next, in Chapter 4, we present the energy-area trade-off of interconnect design techniques optimized based on traffic characteristics. Following that, in Chapter 5, we present an integer linear programming (ILP) framework that exploits value characteristics, to design a static encoding schemes optimized for multiple interconnect design objectives, such as energy, performance, and temperature. Then, in Chapter 6, we extend the ILP framework to design dynamic encoding scheme that addresses multiple interconnect objectives. In Chapter 7, we look at the design and impacts of adaptive encoding scheme on interconnect design metrics. Finally, in Chapter 8, we conclude and present directions for future work.

Chapter 2

A Survey of Interconnect Design Techniques and their Impact on Design Metrics

As technology scales, operating frequencies get higher, and die size increases, due to increased integration, the effects of on-chip interconnects has become a limiting factor in overall system performance. Hence, performance of on-chip interconnects has become increasingly important. Though, circuit performance improves with each new chip generation, higher level of integration brings more and longer wiring onto a chip. Hence, the on-chip wire delay has become even more significant portions of the total chip delay, than before. Moreover, interconnect energy, crosstalk, and reliability degrades the performance of circuits. Improvements in technology, circuit design and architectures have been successful in addressing these limitations, to some extent. However, interconnect continues to be the major bottleneck in overall system performance.

The metal interconnect is the most elementary component in modern integrated circuit and has been of interest. Layout and circuit design techniques such as buffer insertion, and wire shaping/sizing addressed interconnect delay. While other techniques such as bus

encoding, charge sharing/recycling, adiabatic logic, low swing differential signaling, etc. were proposed to address other interconnect design metrics, such as energy, temperature, and reliability, which have come into sharper focus in recent years. Wire aspect-ratio increased to compensate for decrease in wire resistance, due to narrower wires, has led to taller wires with more pronounced inter-wire capacitive effect known as capacitive crosstalk or crosstalk which has become the dominant factor in interconnect design. Techniques such as wire spacing, shield insertion, and ordering were proposed to specifically address interconnect capacitive crosstalk, while previous techniques were also extended to address crosstalk. Though most of these techniques were proposed to address a single interconnect design metric, such as delay or energy, they have significant impact on other design metrics.

While surveys of energy efficient interconnect design techniques exist [28, 29], in literature, none of them provide a comprehensive view of all interconnect design techniques their impact on design metrics. In this chapter we describe various interconnect design techniques, their advantages and limitation, and how each technique impacts interconnect design metrics. The first few techniques address interconnects delay, while the remaining techniques address interconnect energy and/or delay. Though, all the techniques were introduced in the context of addressing a specific design objective, most have a significant impact on other design objectives. Table 2.1 provides an overview all the existing techniques and how they impact design metrics.

2.1 Buffer Insertion

Buffer insertion technique was one of the earliest technique proposed to address interconnect delay. This circuit design technique places buffer along the length of interconnect to reduce the quadratic dependence of delay on wire length to a linear dependence [30, 31]. However, buffer insertion technique addresses interconnect delay at the expense of increased power consumption. In addition, insertion of repeaters has an area overhead due to placement

Metrics → / **Techniques ↓**	Delay	Energy	Temperature	Noise	EM	Area
Wire Spacing	$C\downarrow$	$C\downarrow$	$E\downarrow$ $C_T\downarrow$ $R_T\downarrow$	$C_C\downarrow$	$T\downarrow$	X
Wire Shaping	$R\downarrow$ $C\downarrow$	$C\downarrow$	$E\downarrow$ $C_T\downarrow$ $R_T\downarrow$	$C_C\downarrow$	$T\downarrow$ $W\downarrow$ $j\downarrow$	X
Data Encoding	$C\downarrow$	$C\downarrow$ $N\downarrow$	$E\downarrow$	$\frac{di}{dt}\downarrow$ $C_C\downarrow$	$T\downarrow$ $j\downarrow$	
Voltage Swing	$V_{DD}\uparrow$	$V_{DD}\downarrow$	E$\downarrow$	X	$T\downarrow$ $j\downarrow$	X
Current Sensing	$V_{DD}\uparrow$	$V\downarrow$	$E\downarrow$	X	$T\downarrow$ $j\downarrow$	X
Current Sharing		$V_{DD}\downarrow$	$E\downarrow$	$\frac{di}{dt}\downarrow$	$j\downarrow$	
Multi-V_{DD}	$V_{DD}\uparrow$	$V_{DD}\downarrow$	E$\downarrow$		T$\downarrow$	
Multi-V_T	$V_T\downarrow$	$V_T\uparrow\downarrow$	$E\downarrow$		$T\downarrow$	
Buffer Insertion	$V_{DD}\uparrow$	X	X	$C_C\downarrow$	$j\downarrow$	X
Decoupling cap.				$\frac{di}{dt}\downarrow$ IR$\downarrow$		X

Table 2.1: Summary of interconnect design techniques and their impact on design metrics. Adverse impact is denoted with an X while favorable impacts are denoted by the variable and how it is adjusted to improve the design metric. For example, C$\downarrow$ under column delay indicates that a technique reduces line capacitance C to improve delay. The other variables are line resistance R, thermal resistance R_T, thermal capacitance C_T, current density j, supply voltage V_{DD}, threshold voltage V_T, wire width W, resistive noise IR, inductive noise $\frac{di}{dt}$, crosstalk C_C, and electromigration EM.

of buffers. Algorithms were developed for optimal buffers insertion with or without area constraints [32]. Optimal buffer insertion algorithms were also developed for RC tree (or multi-sink topologies) [33, 34, 35, 36]. Nonetheless, these algorithms consider only buffers of uniform size and do not consider cascading or buffer sizing [37].

Buffer (repeater or inverter) insertion methodology to find optimal repeater placement and sizing with crosstalk consideration were proposed [38] to overcome functional failures due to crosstalk noise. Buffer insertion algorithms were also proposed for the RLC interconnect model as inductive effect becomes important at very high frequencies [39]. However, the algorithm for the RLC interconnect does not include capacitive crosstalk which is a dominating effect in interconnect design. In addition, due to energy performance constraints newer architectures operating at lower frequency are more widely used, eliminating the need

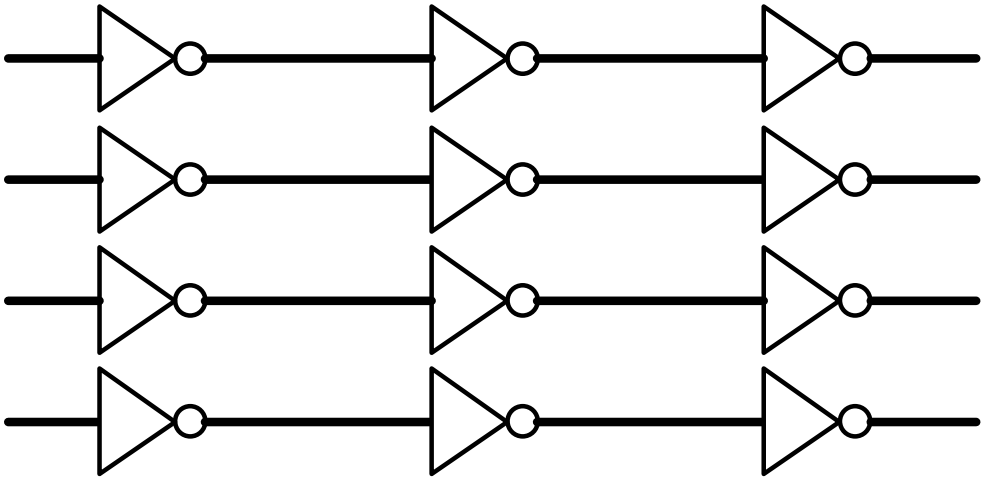


Figure 2.1: Inserting buffers adds more source for supplying power to the interconnect and hence reduces delay. This results in area and energy overhead due to addition inverters.

for complex RLC model.

Buffers layout techniques that interleave buffers of adjoining wires are effective in addressing crosstalk by exploiting the data dependent nature of inter-wire crosstalk. The staggering of repeater effectively limits the worst-case crosstalk by 50% [40]. Another technique, that uses inverting and non-inverting buffers, was also proposed to exploit the data dependent nature of crosstalk to limit worst-case crosstalk while reducing buffer area, power, delay, and placement and delay variation [41]. Buffer circuit techniques such as the SMART buffers [42], skewed repeater delay [43], also reduce worst-case crosstalk by half.

In the staggered repeater configuration, buffer is placed midway between two buffers in the adjacent line. This reduces the worst-case delay to only half of the wire. However, wire delay is not minimized when the buffer is placed at the midway point due to the voltage transfer characteristics (VTC) of the buffer and propagation delay of the wire. Nonetheless, for optimal bus delay the buffers are placed at the midway point [44]. Moreover, optimal buffer placement algorithm have also been used to address other design considerations such as energy [45, 46] and reliability under process variations [47] while meeting delay constraints.

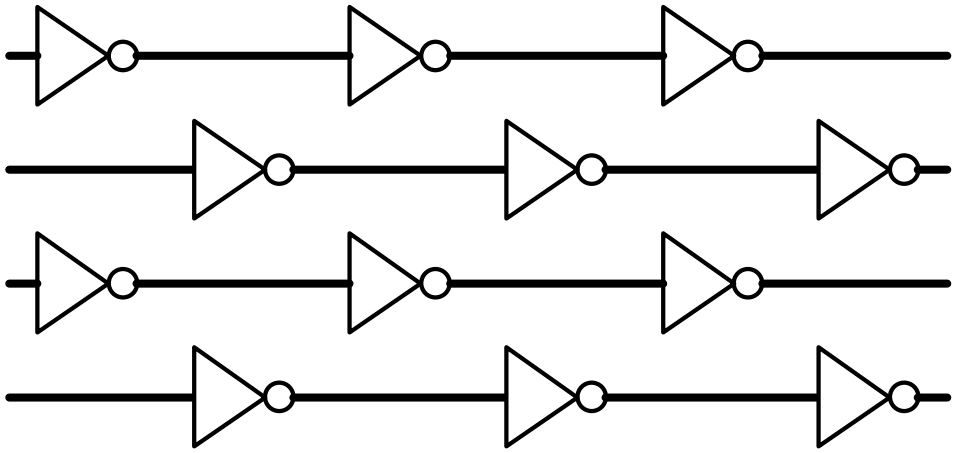


Figure 2.2: Staggering buffers limits the worst-case crosstalk data pattern to half the wire length. This 50% reduction in crosstalk delay improves worst-case wire delay.

2.2 Wire Shaping

Wire sizing or shaping, another technique proposed to address interconnect delay, exploits the distributed nature of interconnection network, by varying the wire width along the length of the wire, to minimize Elmore delay. Early work, divided the interconnect network into small segments and assigned a wire width from one of the possible choices [48, 49] to reduce interconnect delay. Further, closed form expression for continuous wire sizing were derived for optimal delay [50, 51]. However, these expressions are limited due to the discrete nature of design rules. The general minimum delay routing tree is an NP-hard tree, so discrete wire sizing algorithm have been solved optimally for specific topologies (busses) [48], or using heuristics [52, 53, 54, 55] for more general topologies.

Further, the wire sizing problem was simultaneously solved with gate sizing (or buffer insertion) using heuristics to minimize delay [56, 57, 58]. In addition, this problem is solved heuristically for multiple objectives, using a dynamic programming approach for delay-energy optimization [34], and a Lagrange relaxation technique for delay-area optimization [59]. The dynamic programming approach presented for delay-energy optimization was extended to consider process variations [60], also.

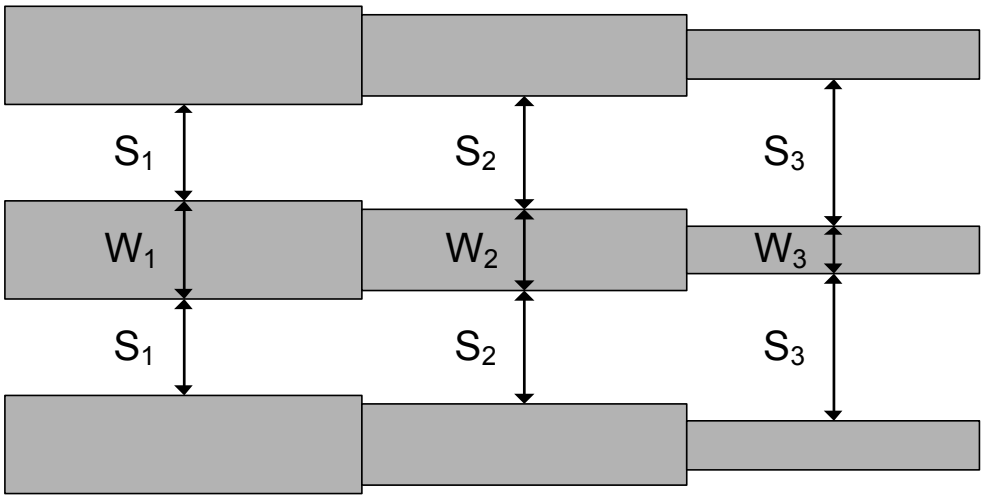


Figure 2.3: The RC delay model shows that current density is higher near the receiver. Hence, wire sizing reduces interconnect delay by reducing wire resistance near the driver by employing wider wires. While, reduction wire width and larger inter wire spacing reduces interconnect energy.

Nevertheless, most of these algorithms considered simple interconnect structure and do not account for coupling capacitances, while most algorithms proposed to account for coupling capacitances assume neighboring wires of uniform width [61, 62] or fixed surrounding [56] to solve the problem, heuristically. Further, wire sizing techniques, which influences wire capacitance, has a profound impact on interconnect energy, because increasing/decreasing width impacts wire's capacitance, and algorithms were proposed to minimize interconnect energy using heuristics [63] or by considering a limited number of discrete wire widths [64], as the original problem was considered NP-hard.

2.3 Data Encoding

With increasing device density and greater integration, energy dissipation has become one of the most important design consideration as it had an unfavorable influence of temperature, performance and reliability of digital circuits. The dynamic energy dissipated due to switching activity accounts for a significant portion of the energy dissipated on interconnects and

the circuit as a whole. So data encoding techniques which transform data from one form to another, as shown in Figure 2.4, were proposed to address interconnect energy.

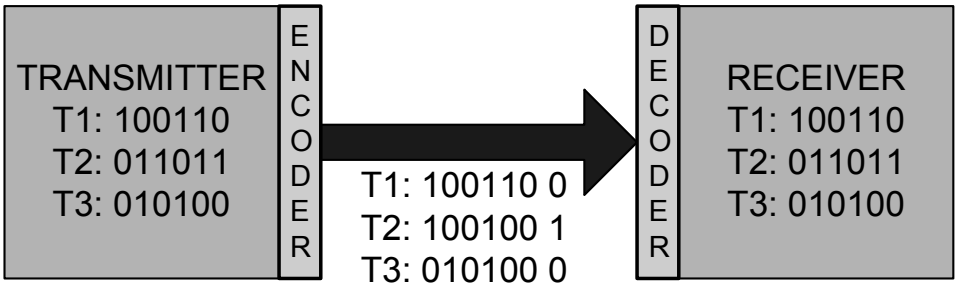


Figure 2.4: Encoding techniques transform the data from one form to another to address interconnect design metrics.

Prior work on encoding can be broadly classified as:

1. **Static encoding schemes**, such as gray code, transition signal coding, etc., transform data, based on traffic characteristics determined at design-time, before transmission to reduce activity on the bus. These schemes have negligible energy, area and power overhead due to encoding circuitry compared to dynamic or memory-based encoding scheme and are particularly effective when bus traffic characteristics are known apriori.

2. **Dynamic encoding schemes** transmits an N-bit word in any cycle using one of (at most) 2^m encoding modes on the N signal wires and indicates the chosen mode to the receiver by setting the value of m extra control wires. Such schemes use transmission metric, like self transition count [65] or coupling transitions [66], to determine how to transmit data at run-time.

3. **Adaptive or Memory-based encoding schemes** employs a cache structure to collect traffic characteristics based on history and determine efficient ways to transmit data. Though it can be safe to assume that encoding overhead for most adaptive encoding techniques are greater than a dynamic encoding scheme, increasing interconnect

length could out weight the advantages of adaptive technique over dynamic or static encoding schemes.

Though most of the works in data encoding address interconnect energy, earlier work addresses performance, area and throughput of buses, especially address buses, through the use of memory-based encoding schemes, such as dynamic base register caching (DBRC) and bus expander (BE) [67, 68]. These techniques used small compression caches at the sending end and register files at the receiving end of a processor-to-memory address bus and were first proposed to reduce off-chip bus widths (area) and pin count. These techniques were subsequently used in compression of on-chip instruction and data streams [69] to improve throughput. These compression techniques were also extended to address interconnect delay [70] and energy [71]. These memory-based encoding were also shown to be relevant in the context of Network-on-Chip [72, 73]. A perspective for optimizing performance, energy, and area/cost of these techniques were shown to yield significant improvements for address busses and are expected to be more pronounced for future technologies [74]. The DBRC was also extended to improve cost and performance, by exploiting value-characteristics, where partial matches in cache rather than a complete match were required for compression [75].

Dynamic encoding schemes such as off-set encoding [76], TO encoding [77], working-zone encoding [78], and irredundant codes [79, 80] was proposed to reduce energy of address buses, while dynamic encoding schemes, such as bus invert (BI) [65], two-dimensional codes [81] and bus invert and transition signaling (BITS) [82], which use self transitions as the metric, and odd even bus invert (OEBI) [66], coupling driven bus invert (CDBI) [83] and coupling based bus invert (CBBI) [84], which use coupling transitions as the metric, have been proposed to reduce energy in data and instruction buses. However, most of the techniques for data and instruction traffic assumes random behavior and yields negligible power savings, for real world traffic such as SPEC CPU 2006 benchmarks that have high redundancy due to temporal, spatial, and value locality in a program [27, 85, 86] and correlation between benchmark programs [87] which are not exploited by the above mentioned schemes.

Previous value-aware dynamic encoding schemes, show the potential of value-aware design approach. Techniques like partial bus invert (PBI) [88] and partition hybrid encoding (PHE) [85], have exploited value aware framework to maximize energy savings by exploiting correlation in real-world traffic. The PBI scheme presents a greedy approach to select a sub-set of wire at design time to be encoded using bus-invert, while PHE splits the bus into disjoint partitions and applies the most energy-efficient dynamic encoding scheme independently to each partition, from among a collection of schemes, using dynamic programming.

However, value-aware dynamic schemes designed using a specific training traffic might increase energy consumption when traffic characteristics differ from the training set. while memory-based encoding schemes, such as adaptive partial bus invert (APBI) [89], frequent value caching [90, 91, 92] and directory based encoding schemes [93], in comparison, dynamically adapt to changing traffic characteristics. The APBI is an extension of PBI where a group of bits are selected at run time based on traffic characteristics and encoded using BI technique. Other memory-based encoding schemes, exploit temporal, spatial and value locality by caching frequently occurring values have been shown to effectively reduce energy for instruction [90, 92] and data traffic [91].

Complex (dynamic and memory-based) encoding schemes reduce bus dynamic energy better than static schemes because they adapt to the traffic. Nonetheless, static encoding schemes, such as bit ordering, that exploit the traffic characteristics to reduce inter-wire coupling activity could be as effective as other complex encoding schemes due to relatively low encoding overhead and low variation in traffic behavior. While bit ordering has been solved using heuristics [94], the problem was shown to be NP-complete and solved optimally using an integer linear programming (ILP) approach [95] and was extended to include a choice of static signaling schemes, such as inverted signaling, transition signaling, etc., where each bit is encoded using one of the signaling scheme [96]. The ILP formulation of the problem allows for multi-objective optimization of interconnect design and has been applied for interconnect energy and temperature optimization using SPEC CPU2K benchmarks [96].

The approach was also evaluated for energy optimal design of area constrained interconnects used in embedded systems [97].

While most encoding scheme have been proposed to reduce bus energy, encoding scheme that exploit data dependent nature of crosstalk have been proposed to improve bus performance. Most of these schemes exploit temporal and spatial redundancy to reduce worst-case crosstalk. Most coding schemes reduce inter-wire crosstalk by 50% [63, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107]. However, simple buffer placement techniques, such as staggered repeater bus, could reduce the worst-case crosstalk by 50% with very negligible circuit and area overhead. Other, techniques such as wire spacing and shield insertion, discussed in the following sections, also eliminate worst-case interconnect crosstalk while improving noise margin. Comparison between various techniques is unavailable in literature. In addition, it is important to note that most encoding proposed to eliminate crosstalk have very high area overhead, due to additional control lines, and might not be as efficient as simpler techniques such as wire spacing or buffer staggering.

Among all the interconnect design techniques presented in this chapter, data encoding techniques are most relevant due to their applicability in future interconnect technology, such as optical interconnect, carbon nanotubes, etc., where circuit and layout techniques, such as buffer insertion, wire spacing and shield insertion are irrelevant.

2.4 Wire Spacing and Shield Insertion

Wire spacing and shield insertion address interconnect delay and energy by reducing the worst-case crosstalk. While wire spacing reduces the inter-wire capacitive coupling, shield insertion eliminates the worst-case crosstalk data-pattern. Shield insertion techniques, simultaneously route signal and power lines to improve interconnect delay and signal integrity [108]. This technique substantially reduces inductive and coupling noise but increases the area and energy overheads. However, wire spacing technique where space between signal

wires is increased to reduce inter-wire capacitance has relatively less area overhead compared to shield insertion [109].

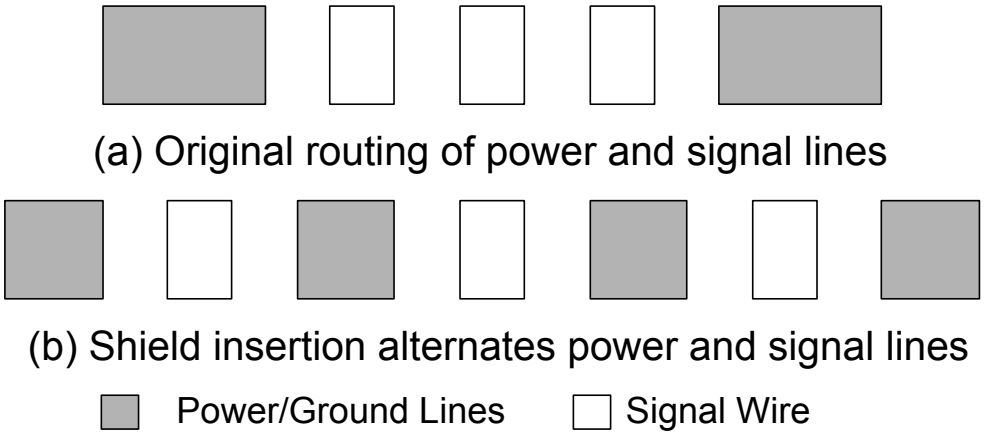


Figure 2.5: Alternating signal and power/ground lines eliminates data dependent crosstalk, provides very high signal integrity and reduces noise. However, there are more signal lines than power/ground lines and introduction of additional power/ground lines results in area overhead and routing congestion.

The optimal wire spacing for crosstalk avoidance was solved using sequential quadratic programming approach [110, 111], while energy minimization through spacing for address buses based on traffic characteristics was solved using heuristics [112]. However, wire spacing problem was shown to be convex and was solved optimally using convex programming [113], optimally. Wire spacing was also solved simultaneously with bit ordering, a static encoding technique, using a greedy approach [114, 115] or by approximating the problem objective [116], to reduce interconnect crosstalk. All these solution provide for continuous wire spacing solution and do not account for the discrete nature of design rules. Discrete wire spacing problems, on the other hand, was solved using dynamic programming for optimal energy and delay objectives.

While all wire spacing problems assume some traffic characteristics at design time, none of them show that design obtained using certain traffic characteristics will perform reasonably well when traffic characteristics change.

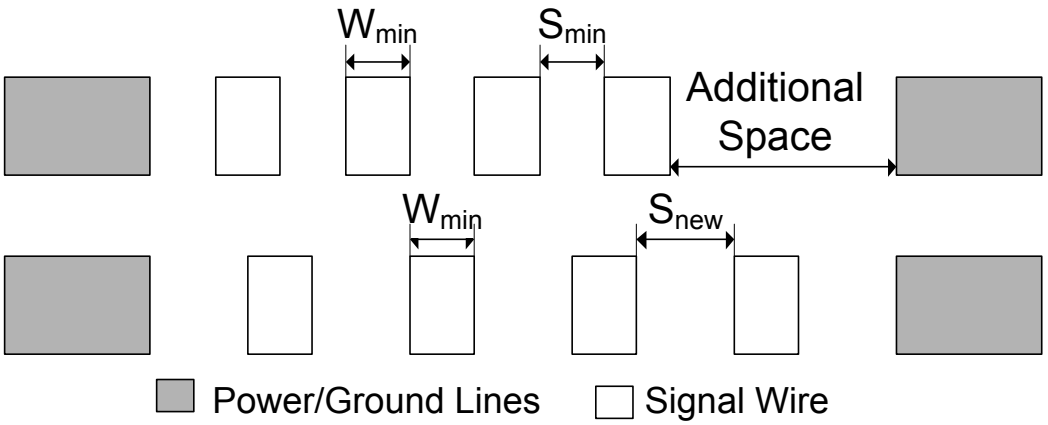


Figure 2.6: Increasing spacing between wire reduces the inter-wire coupling capacitance and improves delay and reduces energy consumption.

2.5 Low Swing or Differential Voltage

Low swing signaling is a bus design technique employed to reduce power dissipation and drive long interconnects. This technique uses two wires to transmit data differentially across the length of the wire and detected by a sense amplifier, similar to an SRAM. This technique increases cost due to the use of two wires instead of one and could use only one clock-phase due to pre-charging cycle.

To overcome the routing overhead, due to the additional wire, single wire low swing signaling have been proposed using symmetric driver and level converter (SDLC) [117], dynamically controlled driver (DCD) [118], low swing bus using charge sharing [119, 120, 121] or pseudo-differential interconnect [122].

2.6 Current Sensing

Current Sensing, where direction of current on the wire is sensed, is an alternative to low-swing voltage sensing for long interconnects because time constant (L/R) is independent of length and limited by device switching speed and time of flight. Hence, current sensing can mitigate the growing effects of interconnect scaling. However, the technique is susceptible

to noise due to inductively coupled return currents.

Current sensing circuits were initially presented for SRAM memory design [123, 124, 125] and later extended to design of high speed interconnect. Current sensing technique when combined with buffer insertion technique not only improves the noise margin but also reduces delay and power. However, the current sensed interconnect is not suitable for multi-drop interconnect design and is susceptible to electromigration and self-heating [126, 127].

2.7 Adiabatic Circuits

Adiabatic circuits implement reversible logic to conserve energy and must meet two conditions: a) never turn on a transistor when there is a potential difference between the gate and the source and b) never turning off a transistor when current is flowing through it. Some of the adiabatic circuit families are: 1) Split-Level Charge Recovery Logic (SLCRL) [128], 2) Efficient Charge Recovery Logic (ECRL) [129], 3) Positive Feedback Adiabatic Logic (PFAL) [130], and 4) 2N-2N2P [131]. Most of these techniques require continuously changing source voltage and very long cycle time to achieve zero energy dissipation. However, some of these principles can be applied in the design of partially adiabatic circuits that meet performance constraint while reducing energy dissipation.

2.8 Charge Sharing/Recycling

Charge recycling/sharing is based on the principles of adiabatic logic circuits where charge is transferred from one node to another. These schemes combined with low-voltage swing interconnect could dramatically reduce interconnect energy while meeting performance constraints [119, 121, 132]. More practical charge recovery circuits have been implemented for buses, consisting of multiple bits lines where charge is transferred from one bit-line to another with minimal losses, thereby reducing energy consumed [133]. This was later extended to account for energy due to coupling capacitance [134]. In addition, charge recovery scheme

seemed to yield better results than low-power bus encoding techniques for real-world traffic, such as mediabench benchmarks [135].

2.9 Decoupling Capacitance

Decoupling capacitors are used as temporary current source and low passes for ac signals to reduce voltage fluctuations in power delivery systems (PDS) due to simultaneous switching [136] while improving the performance of the system. However, high leakage current and cost (due to die area) are a limiting factor in the use of decoupling capacitance. Hence, algorithms for optimal sizing and placement of decoupling capacitance have been proposed. Optimal sizing and placement of decoupling capacitance was shown to be a non-linear optimization and has been solved by approximating the objective to a quadratic programming approach [137, 138] or linear programming [139]. The problems were also solved using approaches such as conjugate gradient method [140], and divide and conquer [141]. Charge based decoupling capacitance estimation have been proposed in the context of noise aware floor-planning [142, 143].

2.10 Multi-V_{DD}/V_T technique

The use of buffers to reduce interconnect delay has made supply voltage V_{DD} and threshold voltage (V_T) an important interconnect design parameter. Reducing V_{DD} has a quadratic effect on dynamic energy consumption but lowers the speed of CMOS circuits. However, the drop in buffer speed is slower than power consumption and reducing sub-threshold voltage V_T at the same rate as V_{DD} could help meet performance requirements. However, reducing V_T increases the sub-threshold leakage current (I_{off}), exponentially, leading to increased leakage energy and poor noise immunity. Hence, digital circuits operate using high-V_{DD} and low-V_T to meet delay constraints while switching to low-V_{DD} and high-V_T when timing slack is available or during idle cycles. Nonetheless, such systems require additional routing space

for supply lines, causing routing congestion, and extra masks for additional V_T, increasing fabrication cost. In addition, control logic to determine the mode of circuit operation and voltage regulators to help low-V_{DD} gate drive a high-V_{DD} gate negate some of the advantages of a multi-V_{DD}/V_T system.

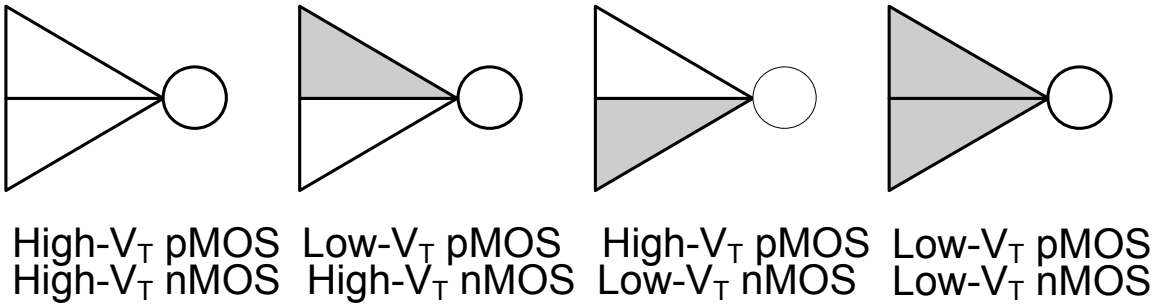


Figure 2.7: High threshold voltage (V_T) reduces leakage energy but increase delay. Dual-V_T buffers reduce leakage energy when output is high or low with minimal impact on delay.

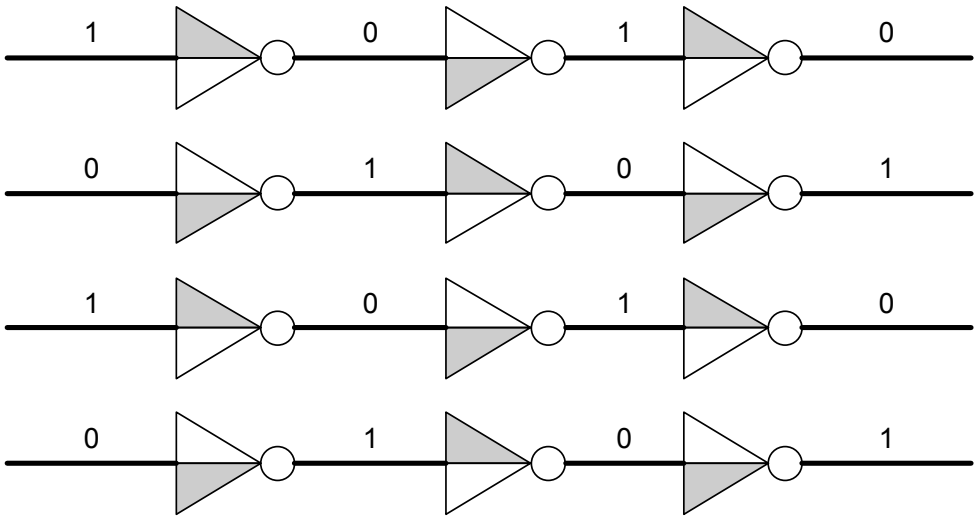


Figure 2.8: Interconnect design to reduce leakage energy for frequently transmitted pattern of 1010.

The problems with regard to multi-V_{DD}/V_T systems have been addressed by assigning a V_{DD} and V_T to a gate, from a set of possible options, at design time. This assignment prob-

lem was solved using a greedy algorithm based on extended clustered voltage scaling [144]. This algorithm was later extended to include gate sizing and solved using an iterative procedure [145]. The V_{DD}, V_T, and gate sizing assignment problem was solved simultaneously using an integer linear programming (ILP) approach [146].

Nonetheless, a major concern for interconnect is the increase in leakage energy in buffers due to low-V_T. This problem was addressed by the use of dual-V_T inverter that utilizes a high-V_T nMOS and low-V_T pMOS or vice-versa. These inverters reduced leakage when output is high (1-state) or low (0-state), respectively [147]. The dual-V_T inverter is designed for each individual wire based on the most frequent value transmitted on that wire, thereby reducing leakage. In addition, this technique could be combined with other techniques that reduce standby leakage, such as the use of an input vector control.

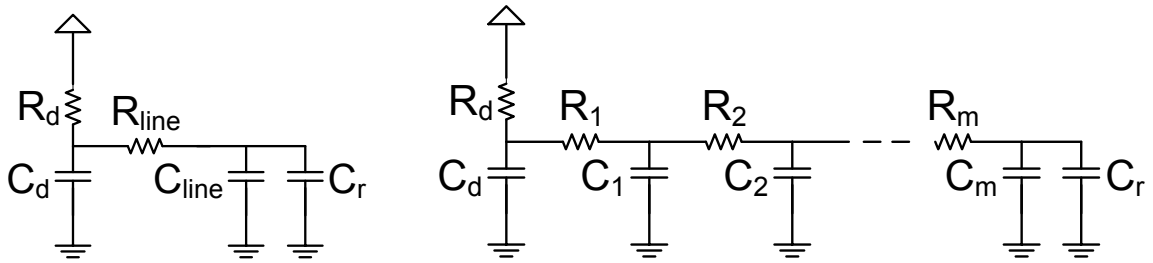
Chapter 3

Interconnect Models and Simulation Methodology

Interconnect analysis applied to performance, power, and reliability considering the effects due to repeaters, delay and slew at the receiver and the effect of switching due to neighboring nets can be performed using dynamic circuit simulation, in which specific stimuli are applied to the circuits and interconnect in question. However, this technique cannot be practically applied to the millions of transistors on a digital IC. Hence, interconnect analysis is performed using simpler models which use the three basic electrical characteristics: resistance (R), capacitance (C) and inductance (L), that depends on interconnect geometry and its position relative to its surrounding structures. Interconnect parasitic i.e., R, L, C, affect circuit performance, capacitance adds load to driving gates, resistance, inductance, and capacitance add signal delay, and inductive and capacitive coupling between interconnect add signal noise.

The *distributed*-RLC interconnect model, compared to the *lumped*-RLC model, is the most accurate approximation of the actual interconnect behavior. Inductive component starts influencing signal propagation delay and transient response, as circuit switching speed increases. However, even in nanometer-scale technologies, particularly for global lines that are longer than 10 mm, it has been shown that the signal response is over-damped when the

line is modeled using the complex distributed RLC model. This implies that interconnects can be modeled using simpler capacitive (RC) model, shown in Figure 3.1, without significant error [148].



(a) Lumped bus model (b) Distributed bus model

Figure 3.1: The distributed RC model where wire is divided into m equal segments and $R_1 = R_2 = \cdots = R_m = \frac{R_{line}}{m}$. R_d and C_d are the driver resistance capacitance respectively, and C_r is the load capacitance due to the receiver.

As technology scales, resistance of interconnect increases exponentially, while that is not the case with inductance and capacitance. Hence, long on-chip interconnects over a few mm in length, and having significant resistance, are adequately represented using a *distributed*-RC model, instead of the pessimistic *lumped*-RC model (or π-model) where the total resistance of each wire segment is lumped into a single resistance R and capacitance C. The *distributed*-model is represented as an m-step π-ladder where the resistance and capacitance of each segment is given by $\frac{R_{line}}{m}$ and $\frac{C_{line}}{m}$, respectively. The accuracy of the analysis increases as m increases.

We assume a standard bus model, as shown in Figure 3.2, comprising a sequence of N+2 parallel, minimum-width, identically-dimensioned, coplanar wires $\langle W_{N+1}, W_N, \cdots, W_1, W_0 \rangle$ from left to right, where $W_1, W_2, \cdots, W_N$ are N signal lines and W_0 and W_{N+1} are two shield (power or ground) lines, with minimum-permissible spacing between adjacent wires [65, 122, 86].

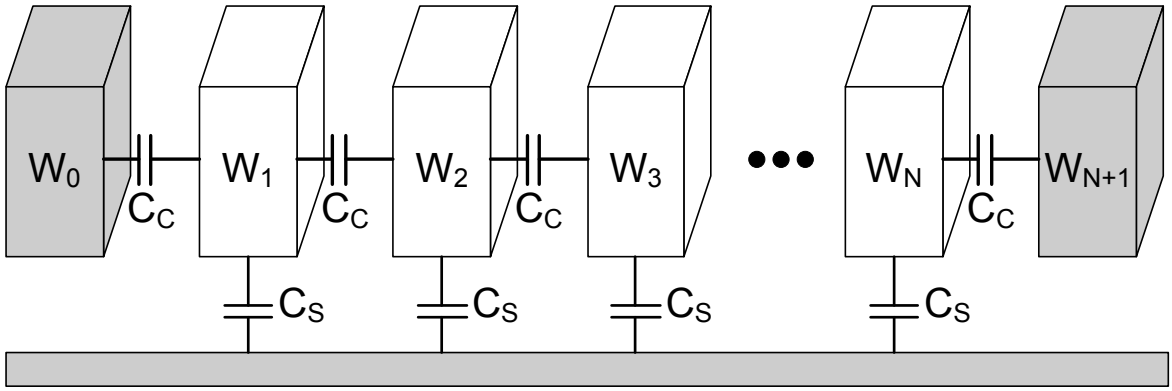


Figure 3.2: A bus consisting of N signal lines, $W_1, W_2, \cdots, W_N$ and two shield line W_0 and W_{N+1}. The shield wires are ground or power line with constant voltage.

3.1 Interconnect Energy Model

3.1.1 Dynamic Energy

A *self transition* occurs when the bit value of a wire *rises* (0→1) or *falls* (1→0), causing the wire's *self capacitance* of value C_S to charge or discharge, respectively, and dissipate self energy of amount $\frac{1}{2}C_S V_{DD}^2$, where V_{DD} is the supply voltage. Then, the total self energy dissipated on a wire is given by Equation 3.1, where $N_S(i)$ is the number of self transitions on i^{th} wire.

$$E_S(i) = \frac{1}{2}C_S V_{DD}^2 N_S(i) \tag{3.1}$$

For current and future technologies inter-wire coupling activity between adjacent wires, has a significant impact on delay, energy and temperature of a wire. The inter-wire coupling activity can be classified as: (a) *no coupling transition* occurs when the bit values remains unchanged or the bit values on both wires rises or falls causing no coupling energy dissipation (b) a *coupling charge* or *discharge transition* occurs when bit value of one wire rises or falls and the other does not change, causing the coupling capacitance of value C_C between the two wires to charge or discharge, respectively, and dissipate coupling energy of amount $\frac{1}{2}C_C V_{DD}^2$

on the first wire; (c) a *coupling toggle transition* occurs when one wire's bit value rises and the other falls, causing a coupling energy dissipation of $C_C V_{DD}^2$ on each of the wire.

The total coupling energy dissipation on i^{th} wire due to adjacent wire is given by Equation 3.2, where $N_{CD}(i, i+1)$ and $N_T(i, i+1)$ are the number of *coupling charge* or *discharge transition* and *coupling toggle transition*, respectively, due to self transition on i^{th} wire. The number of *no coupling transition*, $N_I(i, i+1)$, has no affect on coupling energy dissipated, in the standard bus configuration.

$$E_C(i, i+1) = \frac{1}{2} C_C V_{DD}^2 N_C(i, i+1) \tag{3.2}$$

$$N_C(i, i+1) = N_{CD}(i, i+1) + 2 \times N_T(i, i+1) \tag{3.3}$$

Hence, total dynamic energy dissipated on a wire is given by Equation 3.4 and the total bus dynamic energy E_D is given by Equation 3.5, where N is the bus width.

$$E_D(i) = E_S(i) + E_C(i, i-1) + E_C(i, i+1) \tag{3.4}$$

$$E_D = \sum_{1 \le i \le N} E(i) \tag{3.5}$$

3.1.2 Leakage Energy

In this section, we will discuss the major sources of static energy and how techniques such as dual-V_T inverter, and input vector control can be used to minimize leakage.

Buffers are inserted in global interconnects to improve delay. However, the dynamic and static energy consumption of buffer increases interconnect energy. The current flowing in a transistor during off-state, known as static current, determines the static energy dissipated by the interconnect. The three major sources of static current are: (a) subthreshold leakage current, (b) gate leakage current, and (c) reverse-biased current. Gate leakage and reverse

biased current are expected to be important in the 45 nm technology and beyond. However, subthreshold leakage current will continue to dominate leakage current and is influenced by two major factors: Weak inversion and drain induced barrier lowering (DIBL). The subthreshold leakage current is given by Equation 3.6 [149].

$$I_{leak} = I_0 e^{q(Vgs - V_t)/nkT} \left(1 - e^{-qV_{ds}/kT}\right) \tag{3.6}$$

where

$$I_0 = \mu_0 C_{ox} \left(\frac{W}{L_{eff}}\right) \left(\frac{kT}{q}\right)^2 e^{1.8},$$

V_T is the threshold voltage, C_{ox} is the gate oxide capacitance, μ_0 is the zero bias mobility, and n is the subthreshold swing coefficient.

Though lowering V_T reduces delay of CMOS inverter, it is apparent that it is not suitable from the stand point of leakage energy dissipation. Gate sizing used to improve delay of high-V_T inverter results in higher dynamic power dissipation which negate most of the leakage energy savings. However, interconnect traffic characteristics indicate that an inverter is in *0-state* or *1-state*, more often than the other. Hence, a dual V_T inverter, with high-V_T pMOS and low-V_T nMOS, could be used when 0-state occurs more frequently at the inverter output thereby reducing leakage energy. In addition, the increase in dynamic energy for the dual-V_T inverter is ten times lower than a high-V_T inverter [147]. Further, dual-V_T inverters could use an input vector control to reduce standby leakage thereby reducing delay and power due to sleep transistors.

The leakage energy E_L for an inverter is given by Equation 3.7, where f is clock frequency, N_0 and N_1 are number of cycles the output is at *zero* or *one* state, respectively, and I_{p-leak} and I_{n-leak} is the leakage current of the pMOS and nMOS, respectively. The type of dual inverter chosen determines the leakage current for the pMOS and nMOS. A high-V_T pMOS

is chosen, for the dual-V_T inverter, if the ratio of $N_0/N_1 > 1$, where the mobility of pMOS and nMOS are assumed to be equal. It should also be noted that an increase in the wire self capacitance is sufficient to capture the increased dynamic energy dissipation due to dual-V_T inverter.

$$E_L = \frac{(N_0 I_{p-leak} + N_1 I_{n-leak})}{f} V_{DD} \tag{3.7}$$

3.2 Interconnect Thermal Model

The thermal model uses the analogy between thermal and electrical quantities where the temperature of a wire corresponds to the voltage and heat transfer rate to current [86]. The wire's ability to retain heat is modeled by its *thermal capacitance* and the ability of the surrounding dielectric to conduct heat away from the wire segment is modeled as *thermal resistance*. These thermal circuit parameters are brought together to form a thermal-RC network, shown in Figure 3.3, for a 5-wire bus.

By using Kirchhoff's current law in electrical circuits in the thermal equivalent circuit, we equate the rate of current flowing into a node to the rate of heat flowing out, to obtain Equation 3.8 and Equation 3.9, where P_i is the instantaneous power dissipated in the i^{th} wire, P_i' is the equivalent power due to the effect of switching activity in lower metal layers and the substrate, and θ_{amb} is the ambient temperature (45 °C or 318.15 °K). The instantaneous or cycle-by-cycle power P_i can be obtained by dividing the energy $E(i)$ by the clock cycle time. However, in our micro-architectural simulations, we record the energy $E(i)$ for a finite interval and then divide it by the duration of the time interval (or length of the trace L) to obtain the power dissipated.

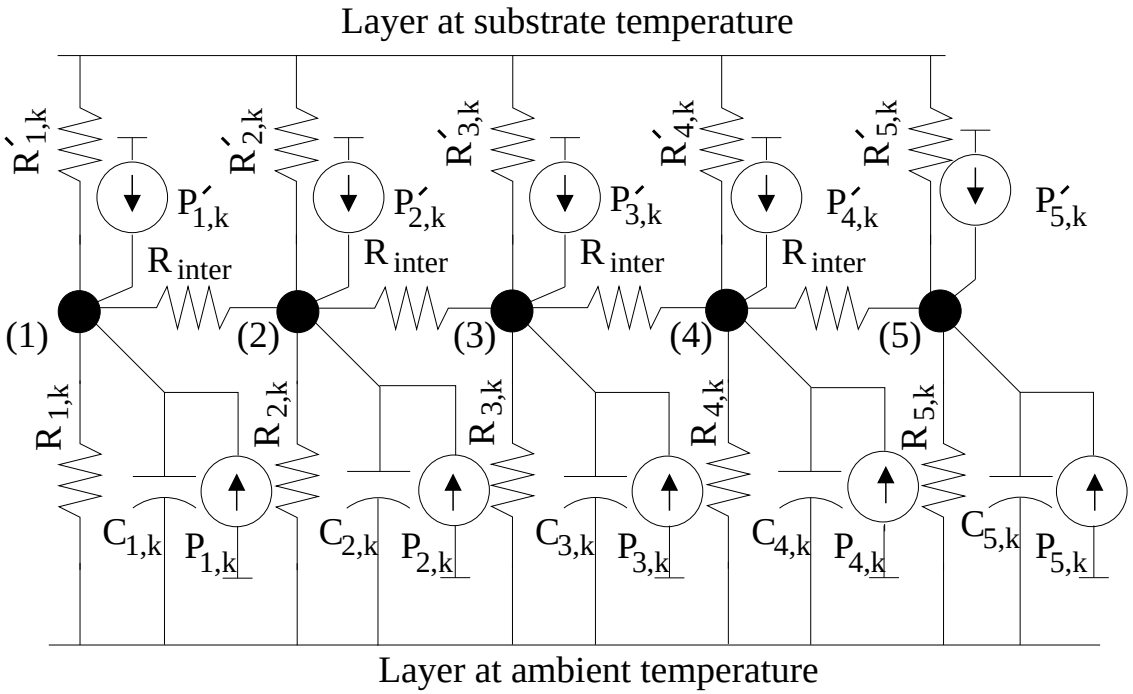


Figure 3.3: Complete equivalent thermal-RC network for a 5-wire bus. $P'_1 = P'_2 = \ldots = P'_5$, $R_1 = R_2 = \ldots = R_5$, $C_1 = C_2 = \ldots = C_5$, and $P_1, P_2, \ldots, P_5$ are bus-activity dependent in the model shown [86].

For the middle wires:

$$P_i + P'_i = \mathcal{C}_i \cdot \frac{d\theta_i}{dt} + \frac{(\theta_i - \theta_0)}{\mathcal{R}_i} + \frac{(2\theta_i - \theta_{i-1} - \theta_{i+1})}{\mathcal{R}_{inter}}, \tag{3.8}$$

and for the two edge wires:

$$P_i + P'_i = \mathcal{C}_i \cdot \frac{d\theta_i}{dt} + \frac{(\theta_i - \theta_0)}{\mathcal{R}_i} + \frac{(\theta_i - \theta_{i\pm1})}{\mathcal{R}_{inter}}, \tag{3.9}$$

In the above equations, the thermal capacitance ($\mathcal{C}_i$), inter-layer thermal resistance ($\mathcal{R}_i$) and inter-wire thermal resistance ($\mathcal{R}_{inter}$) of the bus are determined using wire geometry, inter-layer dielectric constant (k_{ild}) and inter-metal dielectric (k_{imd}) specified by the ITRS [1].

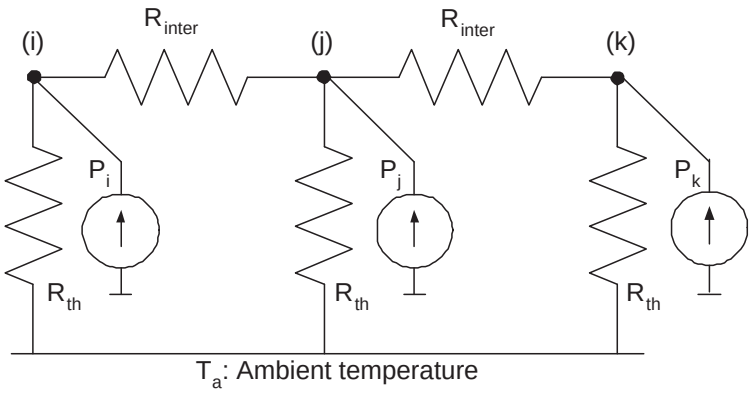


Figure 3.4: Steady state thermal equivalent circuit for three wires. Heat transfer between wires is modeled by R_{inter} and heat loss to surroundings by R_{th}. P_i represents power dissipated in each wire due to switching activity and it can found using a microarchitecture-level simulator [86].

For the middle wires:

$$P_i = \frac{\theta_i - \theta_{amb}}{R_{th}} - \frac{\theta_{i+1} - \theta_i}{R_{inter}} - \frac{\theta_{i-1} - \theta_i}{R_{inter}}, \tag{3.10}$$

and for the two edge wires:

$$P_i = \frac{\theta_i - \theta_{amb}}{R_{th}} + \frac{\theta_i - \theta_{i\pm1}}{R_{inter}}, \tag{3.11}$$

Equation 3.8 can be solved to determine θ_i. However, due to its complexity we use an approximate version of the thermal model, known as *steady-state thermal model* [86], in the temperature optimization methodology. The thermal equations for the steady-state model, shown in Figure 3.4, are given by Equation 3.10 and Equation 3.11 which is solved to obtain wire temperature θ_i, given by Equation 3.12. Thus, we find that the temperature rise ($\Delta\theta_i = \theta_i - \theta_{amb}$) in the middle wire is proportional to a weighted sum of the power dissipated in itself and in its neighboring wires.

$$\theta_i = (P_{i-1} + P_{i+1}) \cdot \alpha + P_i \cdot (\alpha + \beta) + \theta_{amb}, \tag{3.12}$$

$$\text{where } \alpha = \frac{R_{th}^2}{3R_{th} + R_{inter}} \text{ and } \beta = \frac{R_{th}R_{inter}}{3R_{th} + R_{inter}}.$$

3.3 Interconnect Delay and Crosstalk-Noise Model

In high-performance processor buses, inter-wire capacitive crosstalk is the primary factor that affects the propagation delay of interconnects. The data dependent nature of the inter-wire crosstalk necessitates pessimistic worst-case design of bus cycle time, which incurs significant performance penalty as worst-case arises least frequently in actual applications. Hence, activity-aware techniques can significantly reduce the frequency of worst case crosstalk and improve bus performance by using variable cycle bus transmission.

Crosstalk mode	$k-1$, k, $k+1$	Delay factor (g_0)
mode-0	$\uparrow, \uparrow, \uparrow$	$1+0r$
mode-1	$\uparrow, \uparrow, -$	$1+1r$
mode-2	$\uparrow, \uparrow, \downarrow$	$1+2r$
mode-2	$-, \uparrow, -$	$1+2r$
mode-3	$-, \uparrow, \downarrow$	$1+3r$
mode-4	$\downarrow, \uparrow, \downarrow$	$1+4r$

Table 3.1: Bus crosstalk conditions and models for a rising transition in the middle (victim) wire.

The inter-wire capacitance between victim wire and its two adjacent wires dominate the interconnect delay due to capacitive crosstalk. Hence, the simple three-wire interconnect model is more widely used than the more accurate five wire model where the inter-wire capacitance between the victim wire and the two immediate neighboring wires on each side are considered. The five different cross-talk conditions for the three wire interconnect model are presented in Table 3.1, where the coupling ratio r is the ratio of the coupling capacitance C_C and the line capacitance C_S. The coupling ratio for various nanometer technologies are

shown in Table 3.3.

3.4 Other Interconnect Design Metrics

3.4.1 Power Distribution Noise: Inductive and Resistive

Inductive ($L \cdot di/dt$) noise is caused by variations in current across the inductive component of power supply network. As the amount of current drawn fluctuates, the supply voltage fluctuates (rings) proportionally, since $V = L \cdot di/dt$, where V is the supply voltage and i is the current drawn. If this fluctuation exceeds the design noise margins of the supply rail, reliability cannot be ensured. Inductive noise in power distribution is becoming increasingly important with increasing integration and clock frequency. In addition, the resistive noise due to IR drop caused by technology scaling aggravates power distribution noise. As, voltage continues to scale these sources of noise become more critical due to reduced noise margins. Interconnect design techniques, such as data encoding, wire spacing, and shield insertion are effective ways to improving interconnect signal integrity. While data encoding limits the current drawn by limiting the number of transitions on a wire, wire spacing reduces the current drawn by reducing the effective capacitance of the signal wires. On the other hand, shielding reduces the inductive effect by providing closer return path for the current in the signal wire [109].

3.4.2 Interconnect Reliability: Electromigration

Electromigration failure is a major problem in nanometer scale technology. The mean time to failure $MTTF = A\omega j^{-n} e^{Q/kT}$, where A is a constant, j is the current density, k is the Boltzmann constant, Q is the activation energy, ω is the wire width, and T is the absolute temperature [150]. Hence, reliability, due to electromigration, decreases for narrow wires, increased current density, and wire temperature. The ILP framework proposed in this document is used to design encoding schemes that lower peak temperature, which reduces

all stress related breakdowns, including failures due to electromigration.

3.5 Simulation Methodology

In this section we describe our experimental setup and the different optimization scenarios under which value-aware schemes are tested.

3.5.1 Experimental Setup

We used the SimpleScalar/Alpha microarchitecture-level simulator to design and evaluate our technique [151]. The Alpha 21264 architecture modeled by this simulator uses a 64-bit (load/store) data bus between the processor and L1 data cache and a 128-bit instruction bus (fetch width = 4) between the processor and L1 instruction cache. All results reported are for the 32 nm technology node ($R = 3.81$). We used Little-Endian Alpha executables of all 26 benchmarks (12 integer programs and 14 floating-point programs) in the SPEC CPU2000 suite with the `ref` input set and ran our simulations on a shared Linux cluster. We considered bus traffic data corresponding to two representative, non-overlapping 100 M instruction-execution samples based on the SimPoint methodology: a *primary sample* (which is the standard single-simulation point [152]) and a *secondary sample* (which is the early single-simulation point [153]). We selected the SPEC suite as our target workload since pre-compiled Little-Endian executable for our target platform (Alpha 21264) were readily available for it from the SimpleScalar Web site [154]. However, our optimization methodologies are equally applicable to other applications and benchmark suites.

3.5.2 Optimization Scenarios

We also consider three different optimization scenarios with increasing degrees of customization to test our techniques. (1) *General-purpose optimization*: In this case, aggregated traffic transition statistics from the primary samples of thirteen *training benchmarks* (chosen arbi-

trarily to be the set: six integer programs and seven floating-point programs), and then the optimized scheme is tested individually on the secondary traffic samples of the remaining *test benchmarks*. (2) *Workload-specific optimization*: Here, the aggregated traffic transition statistics from the primary samples of all benchmarks were used for optimization, and then tested individually on the secondary traffic samples of the same 15 benchmarks. (3) *Program-specific optimization*: Finally, in this case, for each of the benchmarks, the optimized scheme based on the transition statistics from the primary traffic sample of the benchmark is then tested individually on the secondary traffic sample of the same benchmark. While the first optimization is more relevant in a general-purpose processor design scenario, the latter two are more suitable for particular application domains or embedded systems.

Table 3.2 shows the SPEC CPU benchmarks used for training and testing in the efficacy of our value aware schemes. The abbreviations used in our plots are listed in parenthesis. Though some programs had multiple input files we choose the input files for which single simulation point and early single simulation point were available. The simulation points listed in the table indicated the number of instructions (in 100 million) skipped before SimpleScalar simulation was started.

3.5.3 Technology Parameters

The ITRS roadmap provides wire dimensions, interlevel dielectric constant and other interconnect related parameters for current and future technologies. Hence, to estimate the coupling capacitances between adjacent wires, self capacitance and resistance of a wire we used interconnect model used in predictive technology mode [155]. Using the wire geometry parameters from ITRS (see Table 3.3 for values) we extracted values of self and all coupling capacitances for the middle wire using a 3-wire bus model.

Training Benchmarks			
Benchmark	Input File	SimPoint (in 100 million)	Early SimPoint (in 100 million)
gzip	graphics	653	3
vpr	route	476	71
gcc	200	736	199
mcf	ref	553	316
crafty (craft)	ref	774	0
parser (par)	ref	1146	16
wupwise (wup)	ref	3237	584
swim	ref	2079	5
mgrid (grid)	ref	3292	6
applu (app)	ref	2179	18
mesa	ref	1135	89
galgel (gal)	ref	2491	3150
art	110	340	67
Test Benchmarks			
Benchmark	Input File	SimPoint (in 100 million)	Early SimPoint (in 100 million)
eon	rushmeier	403	18
perlbmk (perl)	makerand	11	1
gap	ref	674	2094
vortex (vort)	two	1024	57
bzip2 (bzip)	graphic	718	51
twolf (wolf)	ref	1066	31
equake (quak)	ref	812	194
facerec (face)	ref	375	136
ammp (amm)	ref	108	2130
lucas (luc)	ref	545	35
fma3d (fma)	ref	2541	298
sixtrack (six)	ref	3043	82
apsi	ref	3408	46

Table 3.2: The input file used for running the program, and the number of instructions (in 100 million) skipped before full system simulation starts are listed in the table. The number of instructions skipped is based on single simulation point or SimPoint methodology.

Year	2010	2012	2015	1018
MPU metal 1 pitch (nm)	45	31.82	21.24	15.02
Number of metal layers	12	12	13	14
Global wire pitch (nm)	240	158	106	72
Global wire thickness (nm)	280	185	124	84
Inter-level dielectric thickness (nm)	240	158	106	72
Inter-level dielectric constant (ϵ_r)	2.3	2.3	2.1	1.9
Self capacitance C_S (fF/mm)	20.39	20.39	18.62	16.85
Coupling capacitance C_C (fF/mm)	63.70	63.89	58.29	52.62
Wire resistance $(K\Omega/mm)$	0.66	1.51	3.35	7.28
Coupling ratio (C_C/C_S)	3.12	3.13	3.13	3.12

Table 3.3: **Technology Parameters:** Wire capacitance and resistance computed using predictive technology model [155] using wire dimensions dielectric constants projected by ITRS [1]. We assume that the inter-level dielectric thickness is equal to the minimum pitch of the global wire.

Chapter 4

Value-Aware Interconnect Energy Optimization under Area Constraints

4.1 Introduction

Efforts to reduce energy dissipation, particularly in long global address, instruction, and data buses, is becoming increasingly important in nanometer-scale technologies as interconnects continue to aggravate performance, power, and cost concerns. More than 50% of the total dynamic power dissipation in a processor is due to interconnects and this is expected to rise to 65%-80% over the next several years based on optimistic interconnect scaling [7]. Increased energy/power dissipation also has an adverse effect on the temperature, reliability, performance, and cost of digital ICs. With technology scaling, the aspect ratio of global interconnects has been gradually increased to check growing resistance and delay of wires relative to those of transistors. However, this causes coupling capacitance between adjacent bus lines to become more pronounced. In current microprocessors, global wires/lines are reported to contribute about 34% of total interconnect power dissipation [7]. Solutions like introduction of low-k dielectrics between global interconnects layers can reduce coupling capacitance to some degree but are not sufficient. Further, introduction of low-k dielectrics

aggravates the problem of wire self-heating [11].

Interconnect design techniques such as bus encoding and wire spacing are widely used to address interconnect energy concerns. Bus encoding is a widely-used approach exploiting the data-dependent nature of bus dynamic energy dissipation in which data is transmitted in an encoded form to save energy. Most existing low-power bus encoding schemes [65, 83, 82, 66, 156], which typically save energy by fully or partly inverting bus data opportunistically in selected cycles require additional control lines to transmit encoded data. In contrast to bus encoding, wire spacing reduces the total inter-wire coupling capacitance by increasing the space between two wires, beyond the minimum required inter-wire spacing. Since, energy dissipated due to coupling capacitance accounts for nearly 80% of the overall interconnect dynamic energy, wire spacing is an effective technique to reduce energy. However, both interconnect design techniques need additional area overhead to address interconnect energy. In addition, both techniques are oblivious to the spatial, temporal and value locality, exhibited by real world programs, such as SPEC CPU 2000 benchmarks [86, 85] and are effective only for random or worse-case (highly-changing) traffic. Moreover, for realistic correlated data streams and, especially, wide (e.g., 64-bit or more) buses, data encoding schemes do not provide appreciable benefits since energy-saving encoding modes are not triggered frequently [86, 85].

Value-aware bus encoding techniques such as partitioned hybrid encoding (PHE) [85], partial bus invert (PBI) [88] were proposed to exploit value characteristics in programs. However, PBI exploits value characteristics only heuristically and considers the impact of self energy, only. While PHE splits the bus into disjoint partitions and applies most energy-efficient encoding scheme from among a collection of schemes to each partition, independently. However, the dynamic programming formulation for the energy-optimal PHE does not limit the area overhead due to additional control lines. On the other hand, value aware wire spacing algorithm, previously presented, based on convex programming [113] is limited by the discrete nature of design rules, while the dynamic programming approach (DP)

presented for value-aware discrete wire spacing [157] can consider only a limited number of spacing due to time complexity of the formulation. Additionally, previous work in wire spacing failed to show that value-aware wire spacing solution, based on certain value characteristics, can work effectively as traffic patterns change during program execution and across different programs.

4.1.1 Key Contributions and Results

We show the efficacy of value-aware design by exploiting traffic characteristics in the design of two widely used interconnect techniques (bus encoding and wire spacing) to minimize interconnect energy. Also, we study the trade-off between energy and area overhead for bus encoding, due to additional control lines, and wire spacing, due increased inter-wire spacing between wires.

In contrast to previous bus encoding techniques, our proposed partitioned hybrid encoding (PHE) technique, splits the bus into disjoint partitions and the most most energy-efficient encoding scheme from among a collection of schemes—we consider bus invert (BI) [65], odd/ even bus invert (OEBI) [66], and not encoding as the possible options, but any other collection of schemes can be considered—is independently applied to each partition for a given control line overhead. Also, we propose an energy optimal greedy algorithm based on convex programming, that provides an energy-optimal discrete wire spacing solution with improved run-time and whose complexity is independent of the number of spacing solutions spacing options considered between any two wires.

The effectiveness of value-aware solution depends on substantial correlation in switching characteristics across program execution and withing different execution regions of a program. To study the impact of value-aware design techniques, under different optimization scenarios, we collected traffic characteristics for data and instruction buses during the execution of real-world programs, represented by SPEC benchmarks, on SimpleScalar/Alpha microarchitectural simulator. PHE and value-aware wire spacing (VAWS) techniques were

optimized for a set of programs (*training benchmarks*) and applied to a different set of programs (*test benchmarks*), known as *general-purpose optimization*, to obtain energy savings of 21.41%/18.89% and 29.55%/24.22%, respectively, for data/instruction bus. The effectiveness of these techniques improves with increasing degrees of customization (suitable for particular application domains or embedded systems) to obtain average bus energy of 21.54%/23.77% and 29.32%/24.88% for workload-specific optimization and 25.43%/28.64% and 30.59%/26.63% for program specific optimization of PHE and VAWS, respectively, for data/instruction buses. To our knowledge this is the first attempt to compare the energy-area trade-off of these two value aware interconnect design techniques.

Next, in Section 4.2, we present the bus energy model followed by related work in Section 4.3. Then, in Section 4.4, we present DP formulation for PHE, given a control line overhead. This is followed by our energy-optimal VAWS algorithm, in Section 4.5. This is followed by a discussion of results in Section 4.6.1 and Section 4.6.2 for PHE and VAWS, respectively. Finally, we conclude in Section 4.7.

4.2 Bus Model

We assume a standard model of a bus comprising a sequence of $N + 2$ parallel, minimum-width, identically-dimensioned, coplanar wires $\langle W_{N+1}, W_N, \ldots, W_1, W_0 \rangle$ from left to right, where $W_1, W_2, \ldots, W_N$ are N signal lines and W_0 and W_{N+1} are two shield (power/ground) lines, with minimum-permissible spacing between adjacent wires [65, 66, 86]. The bus is assumed to use static logic; therefore, it retains a previously-transmitted value until a different one is transmitted. It should be noted, however, that the technique proposed in this paper is equally applicable when dynamic logic is used. The dynamic energy dissipated when an N-bit word is transmitted on the bus in a clock cycle depends upon self and coupling transitions. A *self transition* occurs when the bit value of a wire *rises* ($0 \rightarrow 1$) or *falls* ($1 \rightarrow 0$), causing the wire's *self capacitance* of value C_S to charge or discharge, respectively,

and dissipate self energy of amount $\frac{1}{2}C_S V_{DD}^2$, where V_{DD} is the supply voltage. For a pair of adjacent wires: (a) a *coupling charge* or *discharge transition* occurs when the bit value of one wire does not change and that of the other rises or falls, causing the coupling capacitance of value C_C between the two wires to charge or discharge , respectively, and dissipate coupling energy of amount $\frac{1}{2}C_C V_{DD}^2$; (b) a *coupling toggle transition* occurs when one wire's bit value rises and the other's falls, causing a coupling energy dissipation of $2C_C V_{DD}^2$.

The total bus dynamic energy dissipation in a cycle is therefore $\frac{1}{2}C_S V_{DD}^2[N_S + R \cdot (N_{C+D} + 4 \cdot N_T)]$, where N_S, N_{C+D}, and N_T denote the total number of self, coupling charge + coupling discharge, and coupling toggle transitions, respectively, in that cycle and $R = \frac{C_C}{C_S}$ is referred to as the *coupling ratio*. In current and future nanometer-scale technology nodes, both self and coupling energies are important, although coupling effects are becoming increasingly more pronounced, e.g., $R = 2.082$ for the 130 nm technology node, 2.344 for 90 nm, 2.729 for 65 nm, and 3.051 for 45 nm based on ITRS technology and wire geometry parameters for global interconnects routed in the topmost metal layer [86, 1].

4.3 Related Work

4.3.1 Dynamic Bus Encoding

A number of low-power dynamic bus encoding schemes have been developed. These include schemes, such as bus invert (BI) [65], partial bus invert (PBI) [88], and narrow bus invert (NBI) [82], that seek to reduce self energy (by reducing the average value of N_S) and schemes, such as odd/even bus invert (OEBI) [66], coupling driven bus invert [83], and coupling-based bus invert [156], that attempt to reduce coupling energy (by lowering the average value of $N_{C+D} + 4 \cdot N_T$). In general, a dynamic bus encoding scheme transmits an N-bit word in any cycle using one of (at most) 2^m encoding modes on the N signal wires and indicates the chosen mode to the receiver by setting the value of m extra control wires $\langle W_{c,m-1}, W_{c,m-2}, \ldots, W_{c,1}, W_{c,0} \rangle$. In BI, the encoding modes are: (1) ORG: the original

word as is ($W_{c,0} = 0$) and (2) INV: all bits inverted ($W_{c,0} = 1$). In OEBI, apart from ORG ($W_{c,0}W_{c,1} = 00$) and INV ($W_{c,0}W_{c,1} = 11$), there are two more encoding modes: (3) EVENI: even bits (i.e., those at even-numbered positions in the word) inverted, rest unchanged ($W_{c,0}W_{c,1} = 01$) and (4) ODDI: odd bits inverted, rest unchanged ($W_{c,0}W_{c,1} = 10$).

At the beginning of a cycle, dynamic encoding schemes evaluate a bus energy metric for each encoding mode they support by comparing the current value (for both signal and control wires) on the bus with the new bit pattern to be transmitted, and then choose the mode that minimizes the metric. In BI, the metric is self activity (N_S), and so it attempts to reduce self energy by choosing the mode (ORG or INV) that will minimize N_S in a cycle. In OEBI, the metric is coupling activity ($N_{C+D} + 4N_T$), and so it attempts to reduce coupling energy. When the encoding mode used in a cycle differs from that used for the last transmitted value, the bus transition activity is altered relative to that in the original traffic, and this signifies that an energy-metric-saving mode has been triggered. This, however, does not necessarily mean bus energy is saved if the metric is different from $N_S + R \cdot (N_{C+D} + 4 \cdot N_T)$, which is true for all previous schemes [65, 88, 82, 66, 83, 156].

Thus, in a dynamic encoding scheme, there are three sequential steps processed in hardware at the beginning of every cycle before a word is transmitted: (1) generation of bit patterns for all encoding modes supported, (2) energy metric calculation for these bit patterns, and (3) comparison and selection of the best bit pattern based on the energy metric and the setting of control wire(s) appropriately. This adds to the overall bus latency: the number of logic levels required for these steps is $O(N + m)$ to $O(\log_2(N + m))$ depending upon the hardware topologies (linear or tree, respectively) used for counting (to determine N_S, N_{C+D}, and N_T), metric calculation (e.g., $N_{C+D} + 4N_T$), and comparison (to select the best encoding option).

Among previous schemes, only PBI and PHE bus ecnoding schemes account for value characteristics of the data stream. PBI uses a greedy algorithm to select a subset of, possibly non-contiguous, bus lines and applies BI to it, while the other bus lines remain uncoded.

However, since it uses a heuristic approach, PBI does not fully maximize the frequency of energy-metric-saving modes for BI over the entire bus, and it also does not take into account coupling energy while selecting the BI subset. In contrast, partition hybrid encoding (PHE) [85], splits the bus into disjoint partitions and the most energy-efficient encoding scheme from among a collection of schemes—we consider bus invert (BI) [65], odd/even bus invert (OEBI) [66], and not encoding as the possible options, but any other collection of schemes can be considered—is independently applied to each partition. However, PHE does not limit the maximum number of control lines, which can result in high area overhead. A new DP formulation, presented in Section 4.4, allows us to analyze the trade-off between energy savings and area overhead due to control lines. All other encoding schemes mentioned above were designed for random traffic and/or for narrow buses (8 or 16 bits). As such, they were shown to provide insignificant power savings for wide (64 bit or more) microprocessor buses and correlated traffic, like those found in SPEC CPU2000 benchmarks [86]. This can be attributed to the low frequency with which energy-saving modes were triggered in these schemes.

4.3.2 Wire Spacing

In contrast to bus encoding, wire spacing increase space between wires to reduce (coupling) energy. A number of techniques wire spacing techniques were proposed to minimize delay [158, 159, 160, 161], energy [112, 114, 116], or both [113, 157]. While most of the previous techniques do not solve the problem optimally, the dynamic programming approach [157] solves the problem optimally but has a higher time complexity as it does not exploit the convex nature of the optimization problem. While, the convex programming technique [113] proposed solves wire spacing optimally, but does not account for the discrete nature of wire spacing imposed by design rules. Another major drawback of previous work is the lack of evidence that solutions obtained will work consistently better than uniform wire spacing as workload changes.

In contrast to previous work, the proposed technique produces an energy-optimal wire spacing, under area and discrete spacing constraints. In addition, the time complexity of the greedy approach presented is $\Theta(N \log(N))$, where N is the number of wires considered in the optimization, which is significantly better than the dynamic programming approach [157] despite handling multiple spacing options between wires. The time complexity of the dynamic programming approach is $p^2 N^3 \log(N)$, where p is the number of spacing options.

4.4 Partitioned Hybrid Encoding (PHE)

As we will see in Section 4.6.1, BI and OEBI provide only marginal energy savings on wide buses with correlated traffic; hence, we propose a partitioned hybrid encoding (PHE) technique to optimally partition a bus and apply the most energy-efficient encoding scheme independently to each partition based on traffic value characteristics to minimize total bus dynamic energy. Since PHE is optimized based on traffic value characteristics of one or more programs, its effectiveness on a different program will depend upon how similar their traffic characteristics are. To ascertain this, we first determined the energy dissipated in different bus partitions using the three encoding options we consider in PHE: BI, OEBI, and ORG (i.e., uncoded) across various SPEC CPU2k benchmarks. Then we aggregated bus partition energies for 13 training benchmarks and compared them to the corresponding bus partition energies for remaining benchmarks individually. We found the average correlation between bus partition energies of test benchmarks and aggregated bus partition energies of training benchmarks to be 0.9691/0.9959, 0.9659/0.9958, 0.9700/0.9958 for data/instruction buses for encoding options ORG, BI, and OEBI, respectively. Since the correlation is very close to the ideal value of 1, PHE optimized on one set of benchmarks would indeed be expected to work well on a different set of benchmarks, which is what we find and report in Section 4.6.1.

4.4.1 Bus Partitioning with Isolation Wires

To determine the complete energy of a partitioned bus, we need to calculate the coupling energy between adjacent partitions. In this case, we would need to calculate the coupling activity between the neighboring *edge wires* (leftmost or rightmost wires) of adjacent partitions. The activities of neighboring edge wires of adjacent partitions depend not only on the encoding scheme of the adjacent partitions, but also on the width of those partitions. Hence, the complexity of collecting bus partition energies for a trace containing P N-bit words and a total of s encoding schemes and K encoding modes is $\Theta(s^2N^3P + K^2P)$; this bus partition energy data is needed to optimize PHE.

In order to reduce the complexity of energy data collection to $\Theta(sN^2P + K^2P)$, we designate the signal wires bracketing a partition on its left and right as isolation wires. An *isolation wire* always transmits its bit values in the original uncoded form and is not a part of any encoded partition. Wires W_0 and W_{N+1}, which are power/ground lines, are by definition isolation wires. Hence, if partition $W_{x,y}$ (comprising signal wires W_x through W_y and any associated control wires) is encoded using encoding scheme k and is a part of the final PHE solution, then wires W_{x-1} and W_{y+1} are isolation wires that transmit data in the original form. By using adjacent wires as isolation wires, the coupling energy between the edge wires of the partition and the adjacent isolation wire can be calculated independent of adjacent partitions.

When BI or OEBI is applied to a bus partition, the placement of control line(s) within the partition affects bus partition energy; in the case of OEBI, which considers coupling activity, it also affects encoding decisions. Hence, we considered several control line placement options for BI and OEBI, although in every case control lines are placed between the isolation wires bracketing a partition and signal wires within it. In BI, the single control line is placed either on the LSB or the MSB side of the partition. In the case of OEBI, the two control lines can be placed in six different ways on the two sides of the partition.

4.4.2 Dynamic Programming Based Optimization of PHE

Let $E_k^M(x, y)$, $x \leq y$, denote the energy dissipated in a *bus partition* $W_{x,y}$ when an encoding scheme k with M control lines is applied to the partition; this energy includes all the self energy dissipated on wires W_x through W_y (and on any associated control wires), the coupling energies due to inter-wire capacitances within the partition and the coupling energies due to inter-wire capacitances between isolation wires W_{x-1} and W_{y+1} and wires within $W_{x\cdots y}$. Also, let $E_k^M(x, y) = 0$, when $x > y$. In addition, define $E_{min}^m(x, y) = \min_{\forall k, M \leq m} E_k^M(x, y)$ (i.e., it is the energy dissipation of $W_{x,y}$ with the best encoding option).

We now outline our dynamic-programming (DP) based approach to optimizing PHE. Denote by $W_{x\cdots y}$, $1 \leq x \leq y \leq N$, an energy-optimal partitioned hybrid encoding of the signal-wire set comprising signal wires W_x through W_y given that W_{x-1} and W_{y+1} are isolation wires. Therefore, the overall problem is to determine $W_{1\cdots N}$, i.e., the optimal way to apply PHE to the entire bus—recall that W_0 and W_{N+1} are isolation wires by definition. Let $E_{opt}^m(x, y)$ denote the energy of $W_{x\cdots y}$, when m is control lines allowed are used, this includes all the self energy dissipated on wires W_x through W_y (and on any control wires associated with bus partitions within $W_{x\cdots y}$), the coupling energies due to inter-wire capacitances within $W_{x\cdots y}$, the coupling energies due to inter-wire capacitances between isolation wires W_{x-1} and W_{y+1} and wires within $W_{x\cdots y}$.

First, as mentioned earlier, for some training traffic, we collect energy dissipation data for all possible $\Theta(N^2)$ bus partitions $W_{x,y}$, $1 \leq x \leq y \leq N$, for all encoding options under consideration (ORG and two BI and six OEBI variations based on control line placement) and determine the corresponding $E_{min}^m(x, y)$. Let W_{x-1}, W_z, and W_{y+1}, $x \leq z \leq y$, be three isolation wires for an optimal partitioning of the bus. Then, at least for some $W_{x\cdots y}$, W_z must be an isolation wire. If not, then bus partitioning with isolation wires W_{x-1}, W_{y+1}, and some wire $W_{z'}$, $x \leq z' \leq y$ and $z' \neq z$, would lower bus energy, thus leading to a contradiction. Hence, the problem of optimal partitioned encoding of a bus with isolation wires has an optimal substructure, which suggests that DP might be suitable for it.

$$E^m_{opt}(x,y) = \begin{cases} 0, & x > y \\ \min\limits_{\substack{x \le z \le y \\ 0 \le p \le m}} (E^m_{min}(x,y), [E^p_{min}(x, z-1) + E_s(z) + E^{m-p}_{opt}(z+1, y)]), & x \le y \end{cases} \tag{4.1}$$

The PHE problem was shown to have optimal substructure and can was solved using DP [85]. We reformulated the DP, as shown in Equation 4.1 to limit the maximum number of control lines employed. Limiting the number of control lines increased the complexity of the DP algorithm from $\Theta(N^2)$ to $\Theta(m \cdot N^2)$, where m is the maximum number of control lines allowed.

4.5 Value-Aware Discrete Wire Spacing

In this section, we present our energy optimal value aware discrete wire spacing algorithm. This algorithm exploits convex nature of the problem to present an energy optimal solution.

4.5.1 Energy Optimal Wire Spacing

Value-aware wire spacing algorithm previously proposed [113], exploit the convex nature of the problem. However, the algorithm relies on placing a single line optimally from its neighboring wires, which results poor convergence towards the optimal solution. Since, wire spacing addresses only coupling energy (E_c), the problem is formulated as shown in Equation 4.2, where s_i is the inter-wire spacing and x_i is the coupling activity between wires W_i and W_{i+1}, and A is the constant that captures interconnect length, width, distance from the ground plane, and other technology and design parameters, such as supply voltage, clock frequency, metal resistivity and dielectric permittivity which influence interconnect energy. In the optimization problem with N wires and 2 shield lines there are $N+1$ inter-wire space. Equation 4.3 limits the maximum space available for energy optimization. Therefore, there are N variables in the problem. The minimum spacing design constraint in Equation 4.4

ensures that all wires meet the minimum delay constraint for a design.

Minimize

$$E_c = A.\sum_{i=0}^{N} \frac{x_i}{s_i} \tag{4.2}$$

subject to:

$$\sum_{i=0}^{N} s_i = D \tag{4.3}$$

$$s_i \geq s_{min},\ \forall i \tag{4.4}$$

Differentiating the objective in Equation 4.2 w.r.t. N variables $(s_0, s_1, \cdots, s_{N-1})$, we get

$$A.\frac{x_0}{s_0^2} = A.\frac{x_1}{s_1^2} = \cdots = A.\frac{x_N}{s_N^2} = \nu \tag{4.5}$$

$$\implies s_i = \sqrt{\frac{x_i}{\nu}},\ \forall i, \tag{4.6}$$

where ν is a constant. Substituting Equation 4.6 in Equation 4.3 we obtain:

$$\nu = \frac{(\sum_{i=1}^{N} \sqrt{x_i})^2}{D^2} \tag{4.7}$$

Since, x_i and D are constants, the wire spacing between each wire can be calculated in $\Theta(N)$. To ensure that the minimum spacing constraint, in Equation 4.4, is met we assign $s_i = S_{min},\ \forall s_i < S_{min}$. Then, we re-solve the problem, after dropping all variables for which s_i was less than S_{min}, until minimum spacing constraints are met. This convex programming approach has a faster convergence for optimal wire spacing than previous solution [113], as

it provides a closed form solution.

4.5.2 Wire Spacing Under Discrete Design Rules

Optimal wire spacing solution obtained through the closed form solution is limited by discrete nature of wire spacing due to λ-based design rules, where spacing between any two wire is discrete and an integer multiple of λ, defined by technology. To overcome this problem we employ a greedy algorithm at obtain an energy optimal solution under discrete spacing constraint. In addition, the greedy algorithm exploits the solution obtained from continuous wire spacing to reach the optimal solution faster.

The algorithm first truncates the space allocated between wires such that $s_i = \lfloor \frac{s_i}{\lambda} \rfloor . \lambda$. The remaining space is divided into M discrete spaces of λ width, where $M < N$, and are assigned using the greedy algorithm described in Algorithm 4.1. Since, the problem is convex this initial spacing value, obtained from continuous optimization, helps us reach the optimal solution faster. However, the algorithm reaches the optimal solution even if the initial spacing assigned is s_{min}.

The energy equation is rewritten, as shown in Equation 4.8, to show that the problem can be solved optimally using the greedy approach for wire spacing under discrete spacing constraints. Denoted by $E_{0,N}(K)$ is the energy optimal spacing between wires W_0 and W_N, where $K = \lfloor D/\lambda \rfloor$ is the number of discrete spaces available and $e_i(k_i) \propto x_i \cdot k_i \cdot \lambda/(1 + k_i \cdot \lambda)$ is the coupling energy reduction between wires W_i and W_{i+1} due to $k_i \cdot \lambda$ additional spaces allocated and x_i is the coupling activity, between the two wires.

If $e_x(X)$ and $e_y(Y)$ are part of the optimal solution, then there does not exist a solution $e_x(X')$ and $e_y(Y')$ such that $e_x X' + e_y(Y') < e_x X + e_y(Y)$ and $X + Y = X' + Y'$. If $e_x X'$ and $e_y(Y')$ exists then the overall energy can be further reduced, which is a contradiction. Hence, the problem has an optimal substructure and is formulated as a recursive function, given by Equation 4.9.

$$E_{0,N}(K) = \min \sum_{i=0}^{N} \frac{x_i}{(1 + k_i.\lambda)}, \text{ s.t. } \sum_{i=0}^{N} k_i = K \tag{4.8}$$

$$E_{0,N}(K) = \min_{0 \le k_0 \le K} [e_0(k_0) + E_{1,N}(K - k_0)] \tag{4.9}$$

Since, the problem is convex and has an optimal sub-structure and can be solved using a recursive function, we can solve the problem optimally using a greedy algorithm based on $g_i = x_i \cdot \lambda / s_i \cdot (s_i + \lambda)$, where g_i is the energy saving obtained by adding additional space λ between wires W_i and W_{i+1}, where s_i is the current spacing between the two adjacent wires and λ is smallest increment in space allowed by the design rules. Since adding space λ between wires with $\max g_i$ minimize the energy optimally, the greedy approach described below obtains the energy optimal wire spacing by iteratively adding λ space between wires with $\max g_i$ during each iteration.

Algorithm 4.1 Greedy algorithm based on convex programming.

```
Compute s_i using the convex programming technique
Initial discrete space assigned s_i = ⌊s_i/λ⌋.λ
Energy gained by adding λ space g_i = x_i/s_i * (s_i + λ)
for m = 0 to M − 1 do
    if g_j = max_{∀i}(g_i) then
        s_j = s_j + λ
        Recompute g_j
    end if
end for
```

4.6 Results and Discussion

4.6.1 Partitioned Hybrid Encoding

In this section, we present results on bus dynamic energy savings obtained, accounting for energy dissipated by control lines, using the previous schemes BI and OEBI and compare

them to those obtained using our PHE technique for the SPEC CPU2k benchmarks. All results reported for encoding schemes are in the form of *normalized bus energy*, where energy components for each benchmark is normalized w.r.t. the energy dissipated on the original uncoded bus for that benchmark.

Across the benchmarks, we found the average energy savings for BI to be only 1.85% and 3.28% for the data and instruction buses, respectively. The reason for these meager savings is the relatively small number of self transitions per word transmitted in the original uncoded bus: 11.74 and 11.79 for the data and instruction buses, respectively, far fewer than the $\lceil N/2 \rceil + 1$ transitions required to trigger the inversion mode. In fact, we find that energy-metric saving modes are triggered on the average only 15.58% and 19.20% of the time for data and instruction traffic, respectively. Although energy-metric saving modes were triggered much more often in the case of OEBI—30.49% and 66.48% of the time for data and instruction buses, respectively—average energy savings were only 0.63% and 10.45% for data and instruction buses, respectively. This can be attributed to the much higher self energy caused by the encoding modes chosen by OEBI compared to the coupling energy saved. It should be noted that the energy savings reported for OEBI for instruction traffic is different from results reported previously [85] because we consider a 32-bit instruction bus instead of a 128-bit instruction bus. This result shows the influence of bus width on encoding schemes which do not account for the value characteristics. In addition, PHE energy savings for 32-bit instruction traffic were comparable to previous results.

Compared to BI and OEBI, the proposed PHE technique provides significant energy savings. Results for the three optimization scenarios are given in Figures 4.1, 4.3, and 4.5 for data bus and in Figures 4.2, 4.4, and 4.6 for instruction buses, with energy fractions due to self, coupling charge and discharge, and coupling toggle shown separately. It can be seen that average energy savings obtained on test traffic samples are: 21.41% and 18.89% for general-purpose optimization, 21.54% and 23.77% for workload-specific optimization, and 25.43% and 28.64% for program-specific optimization cases for data and instruction traffic,

respectively. These results are 2-20 times better than those for BI and OEBI, thus underscoring the efficacy of our approach. Energy savings obtained by PHE are fairly uniform across benchmarks, with most of the energy savings arising from a reduction in coupling toggle energy and to some extent in self energy, at the expense of sometimes slight increase in coupling charge and discharge energy. This means that the effectiveness of PHE (incorporating BI and OEBI) is likely to improve with technology scaling as R increases and causes coupling toggle energy to be even more pronounced.

While PHE maximizes the energy savings by exploiting the value characteristics, it does have an area overhead due to additional control lines. The area overhead versus the control line overhead are shown in Figure 4.7 for data and instruction traffic. The number of control line for energy optimal PHE is 18 and 6 representing and area overhead of 28.13% and 18.75% due to control lines for data and instruction bus, respectively. However, we do notice that the energy savings for data bus almost reaches the maximum limit with only 10 control lines (overhead of 15.63%), with each additional control line reducing energy by less than 0.2%.

4.6.2 Value-Aware Wire Spacing

In this section, we present results on bus dynamic energy savings obtained, using value wire spacing and compare them to those obtained using our PHE technique for the SPEC CPU2k benchmarks. The results are reported for area overhead equivalent to 10 and 6 additional lines for data and instruction traffic, respectively. Results for the three optimization scenarios are given in Figures 4.8, 4.10, and 4.12 for data bus and in Figures 4.9, 4.11, and 4.13 for instruction buses, with energy fractions due to self, coupling charge and discharge, and coupling toggle shown separately. It can be seen that average energy savings obtained on test traffic samples are: 29.55% and 24.22% for general-purpose optimization, 29.32% and 24.88% for workload-specific optimization, and 30.59% and 26.63% for program-specific optimization cases for data and instruction traffic, respectively.

The energy savings obtained through value aware wire spacing are significantly better

for data traffic compared to PHE, while energy savings for value-aware energy savings is only slightly better than PHE. In addition, energy savings for PHE and wire spacing are comparable for lower number of control lines, as shown in Figure 4.14, for both data and instruction traffic.

4.7 Conclusion

In this chapter, we have demonstrated the effectiveness of value-aware interconnect design techniques, such as bus encoding and value-aware wire spacing, by applying it to data and instruction bus carrying realistic traffic generated by SPEC benchmarks. In addition, we demonstrated effectiveness of two interconnect design techniques in reducing energy for various area overheads. While energy savings for value aware wire spacing and bus encoding techniques are comparable for low area overhead, wire spacing works better when more area is available for energy optimization because data encoding is limited by the transition activity on the bus. In addition, value-aware bus encoding technique needs to be applied during the early design phase, while value-aware wire spacing can be applied during early design phase and as a post routing optimization technique to utilize white space introduced by CAD routing algorithms [162]. However, designers using wire spacing, should be aware of decreased thermal resistance due to increased inter-metal dielectric and reliability issues associated with increased thermal stress [16, 163]. Further, introduction of low-k dielectrics aggravates the problem of wire self-heating [11]. In addition, wire spacing as an energy minimization interconnect technique is applicable only for metal/dielectric interconnect, while encoding schemes are applicable in the context of carbon nanotubes, optical interconnect, and other future interconnect technologies.

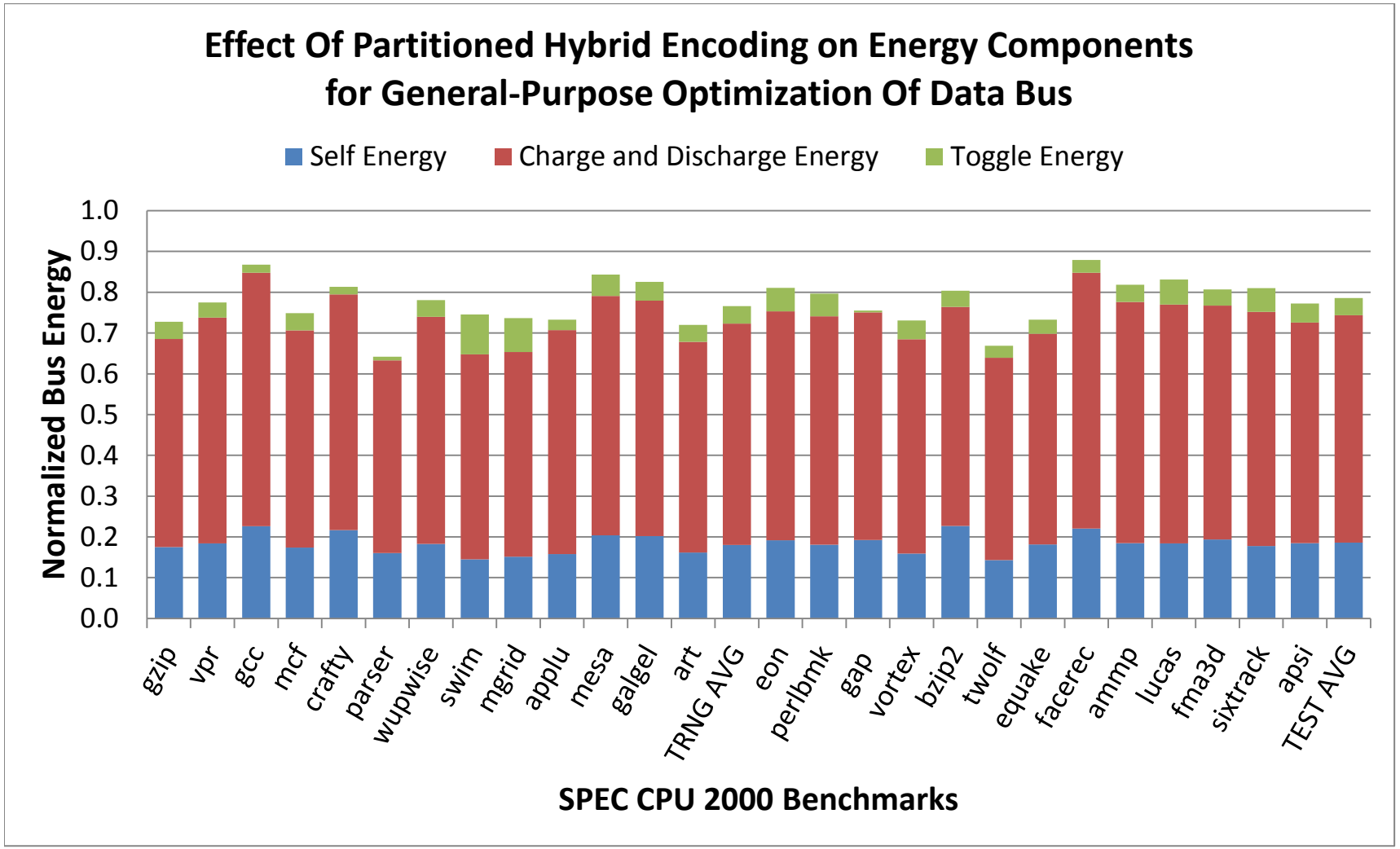


Figure 4.1: **General-Purpose Optimization for Data Bus:** Bus energy of SPEC CPU2K training (TRNG) and test (TEST) programs, due to partitioned hybrid encoding (PHE), normalized w.r.t the original uncoded bus energy.

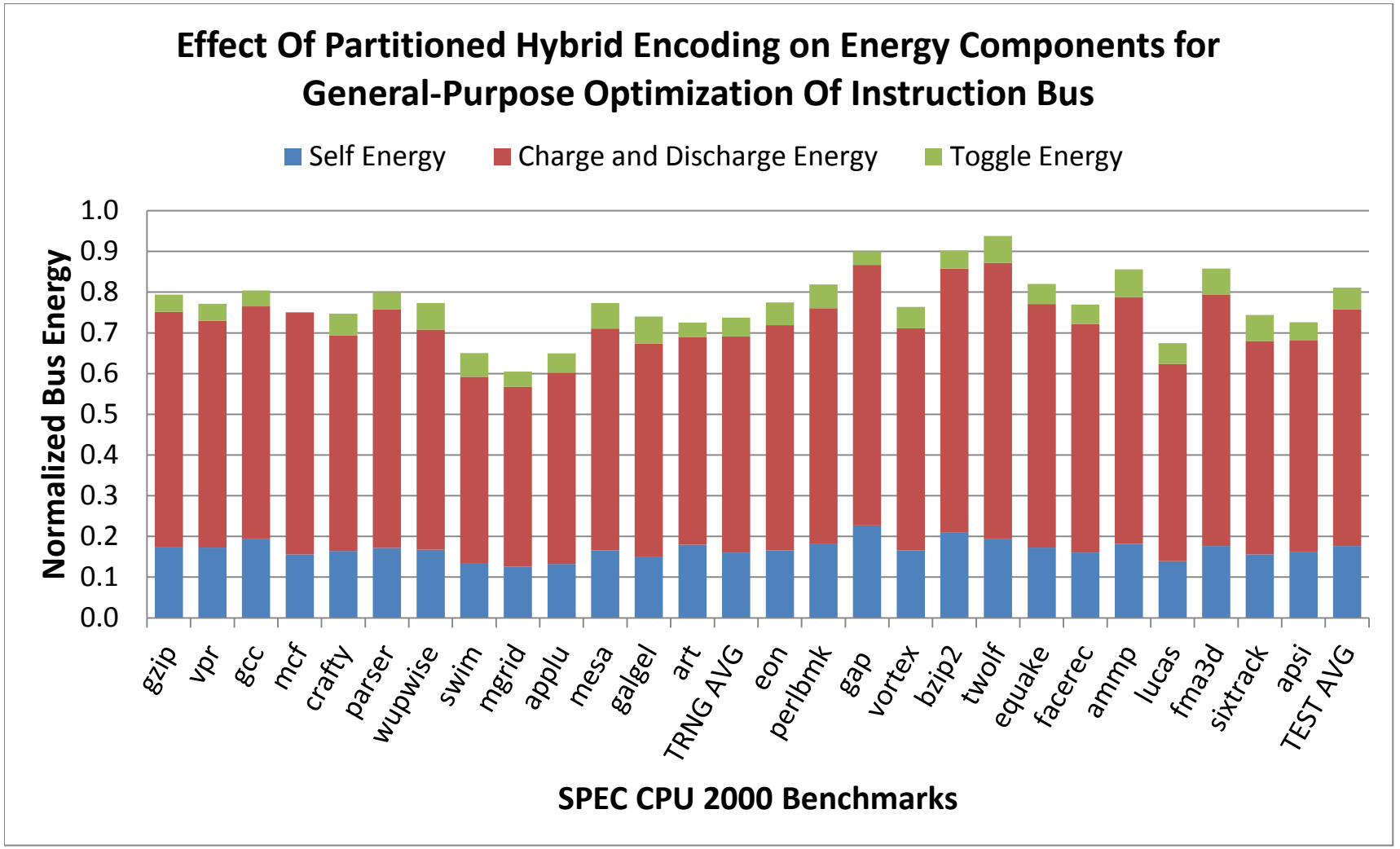


Figure 4.2: **General-Purpose Optimization for Instruction Bus:** Bus energy of SPEC CPU2K training (TRNG) and test (TEST) programs, due to partitioned hybrid encoding (PHE), normalized w.r.t the original uncoded bus energy.

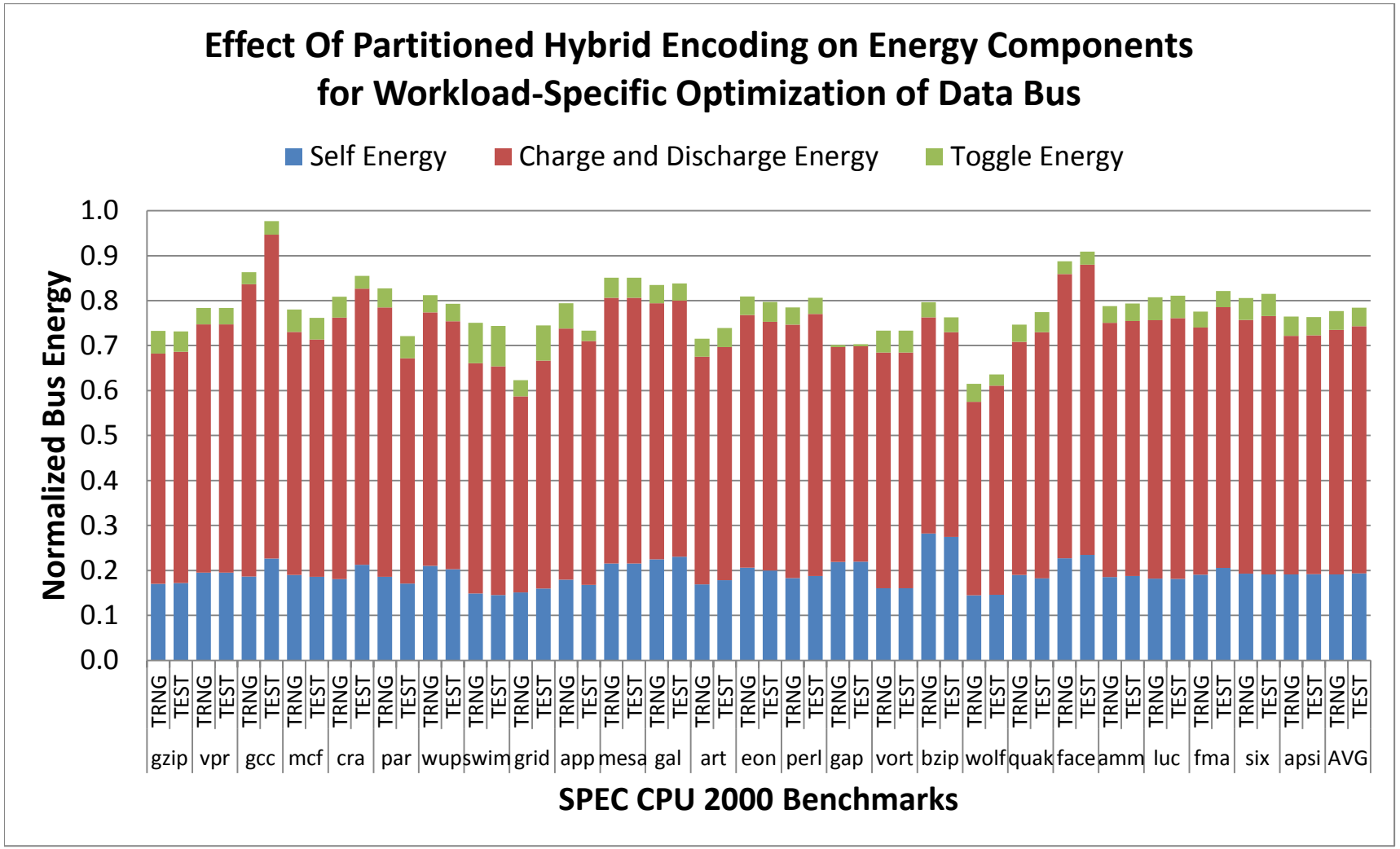


Figure 4.3: **Workload-Specific Optimization for Data Bus:** Bus energy of training (TRNG) and test (TEST) samples of SPEC2K benchmarks, due to partitioned hybrid encoding (PHE), normalized w.r.t original uncoded bus energy.

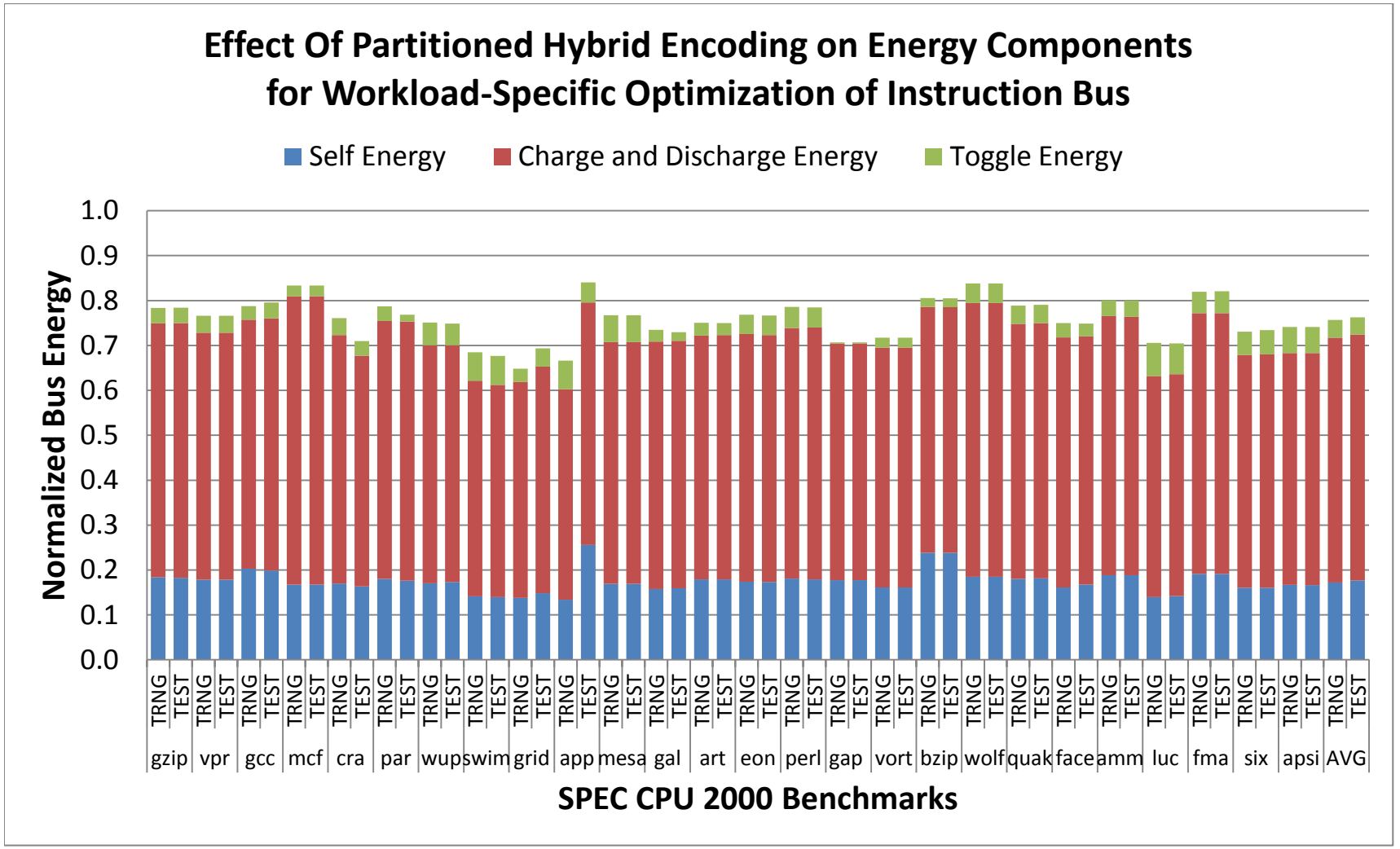


Figure 4.4: **Workload-Specific Optimization for Instruction Bus:** Bus energy of training (TRNG) and test (TEST) samples of SPEC2K benchmarks, due to partitioned hybrid encoding (PHE), normalized w.r.t original uncoded bus energy.

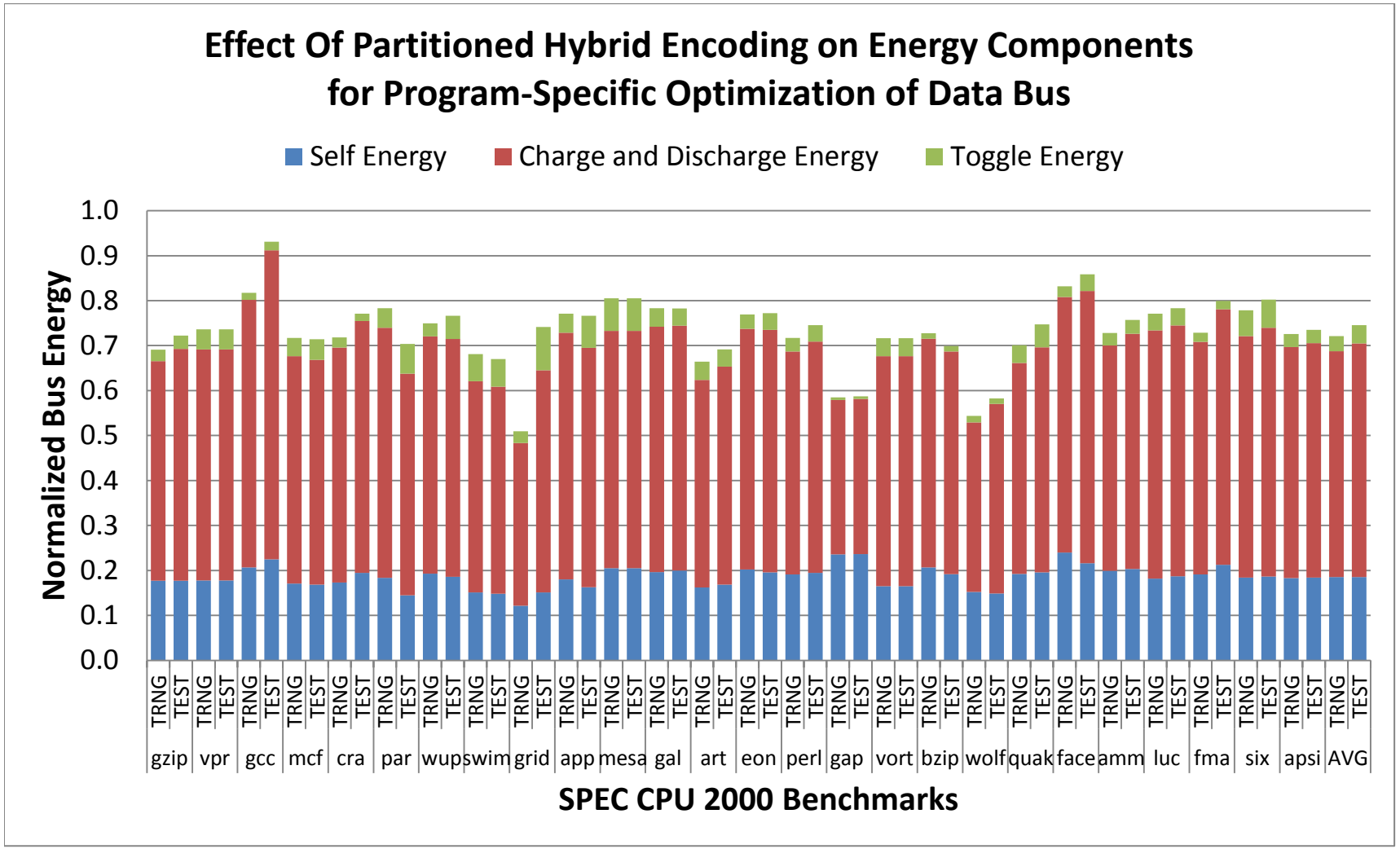


Figure 4.5: **Program-Specific Optimization for Data Bus:** Bus energy of training (TRNG) and test (TEST) samples of SPEC2K benchmarks, due to partitioned hybrid encoding (PHE), normalized w.r.t original uncoded bus energy.

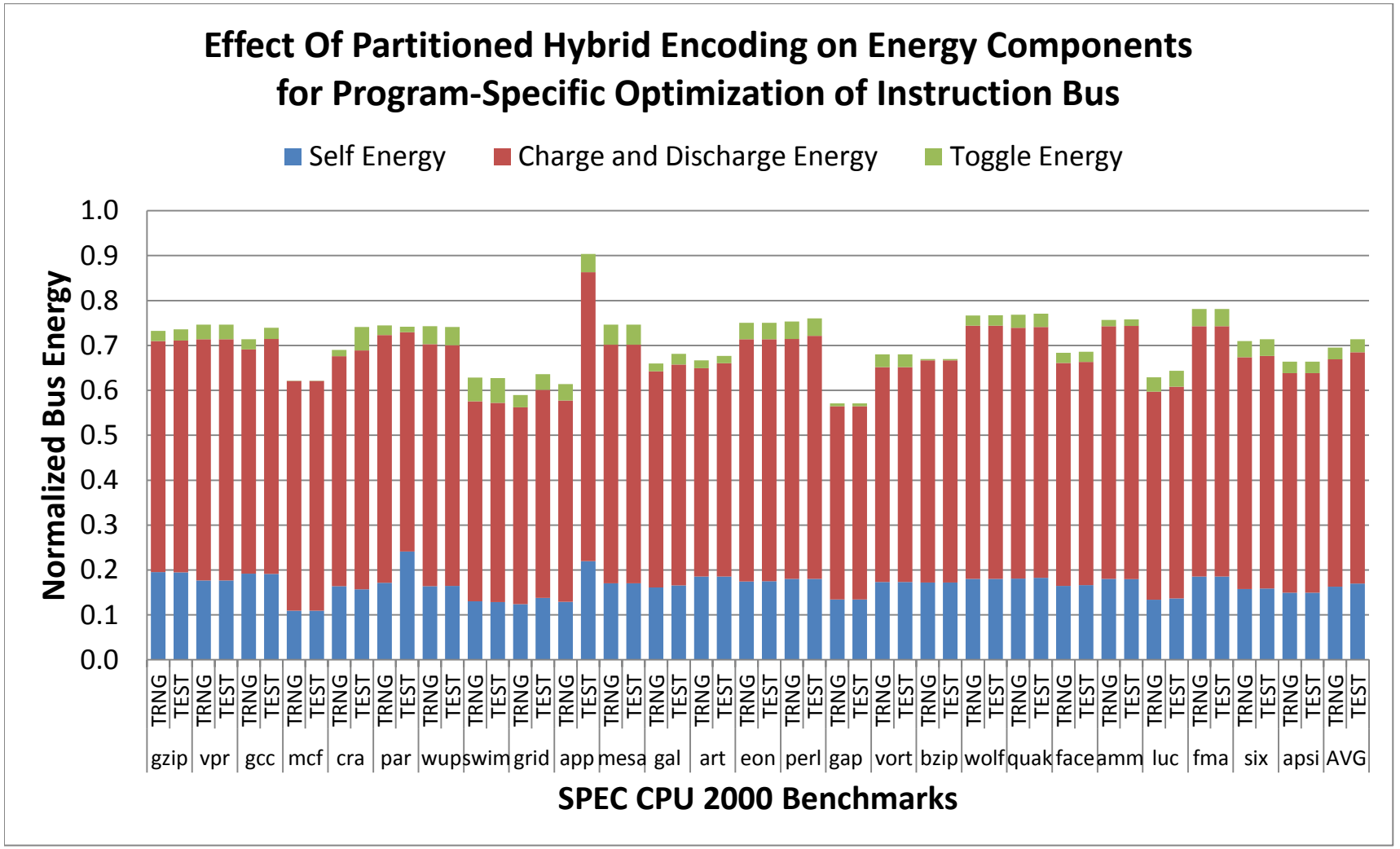


Figure 4.6: **Program-Specific Optimization for Instruction Bus:** Bus energy of training (TRNG) and test (TEST) samples of SPEC2K benchmarks, due to partitioned hybrid encoding (PHE), normalized w.r.t original uncoded bus energy.

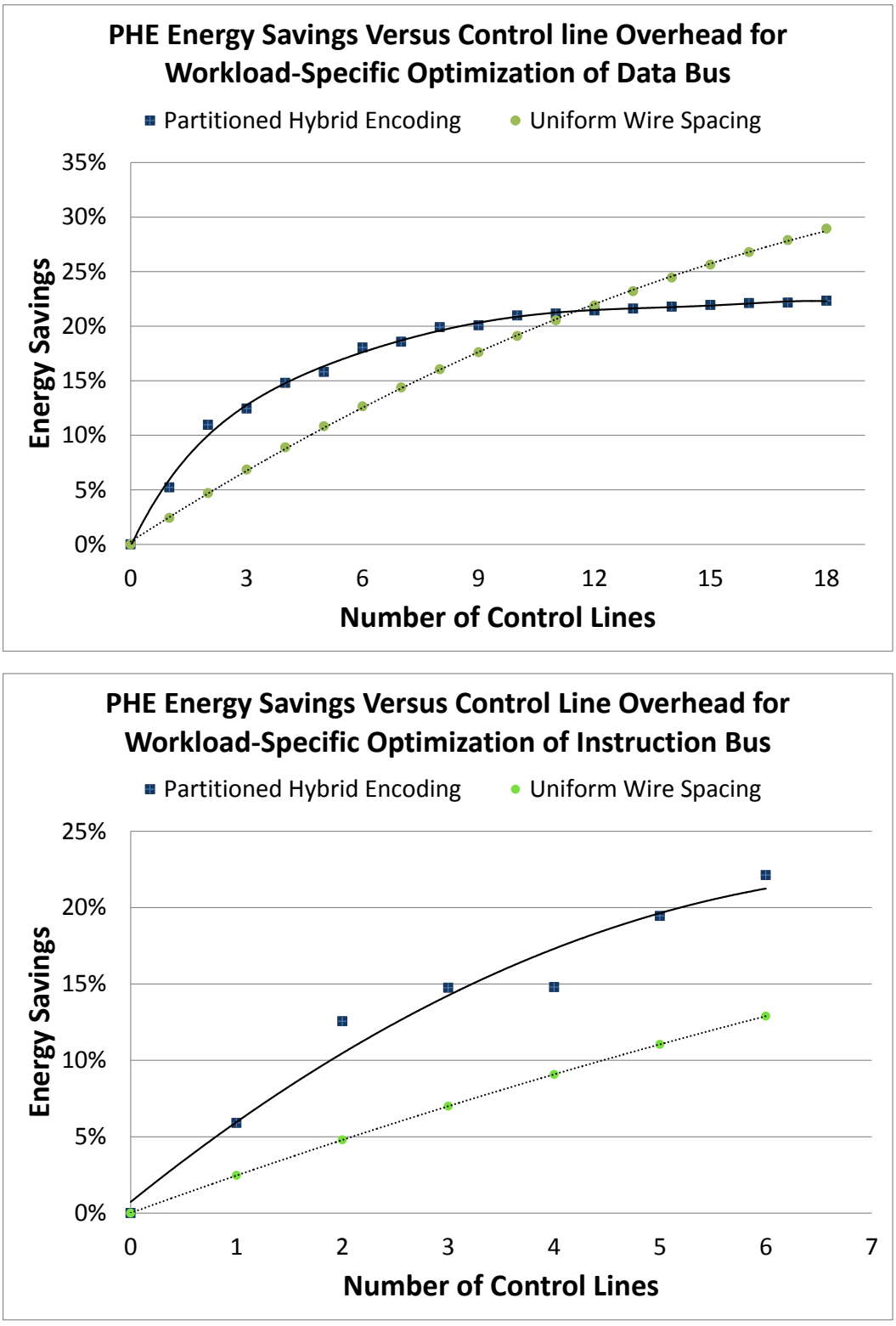


Figure 4.7: Energy savings for partitioned hybrid encoding (PHE) for different area overheads compared to uniform wire spacing for data and instruction traffic, respectively.

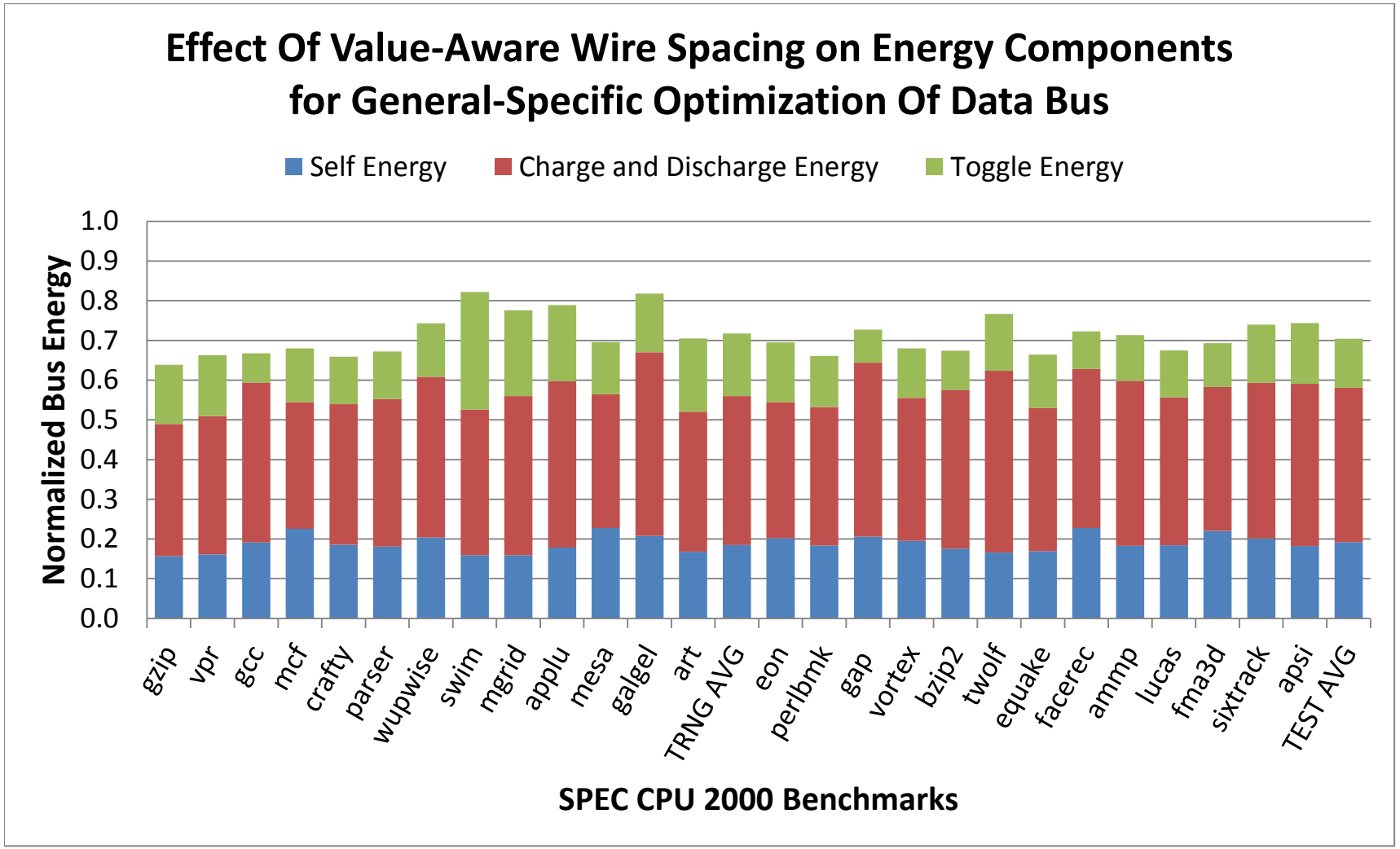


Figure 4.8: **General-Purpose Optimization for Data Bus:** Bus energy of SPEC CPU2K training (TRNG) and test (TEST) programs, due to value-aware wire spacing (VAWS), normalized w.r.t the original uncoded bus energy.

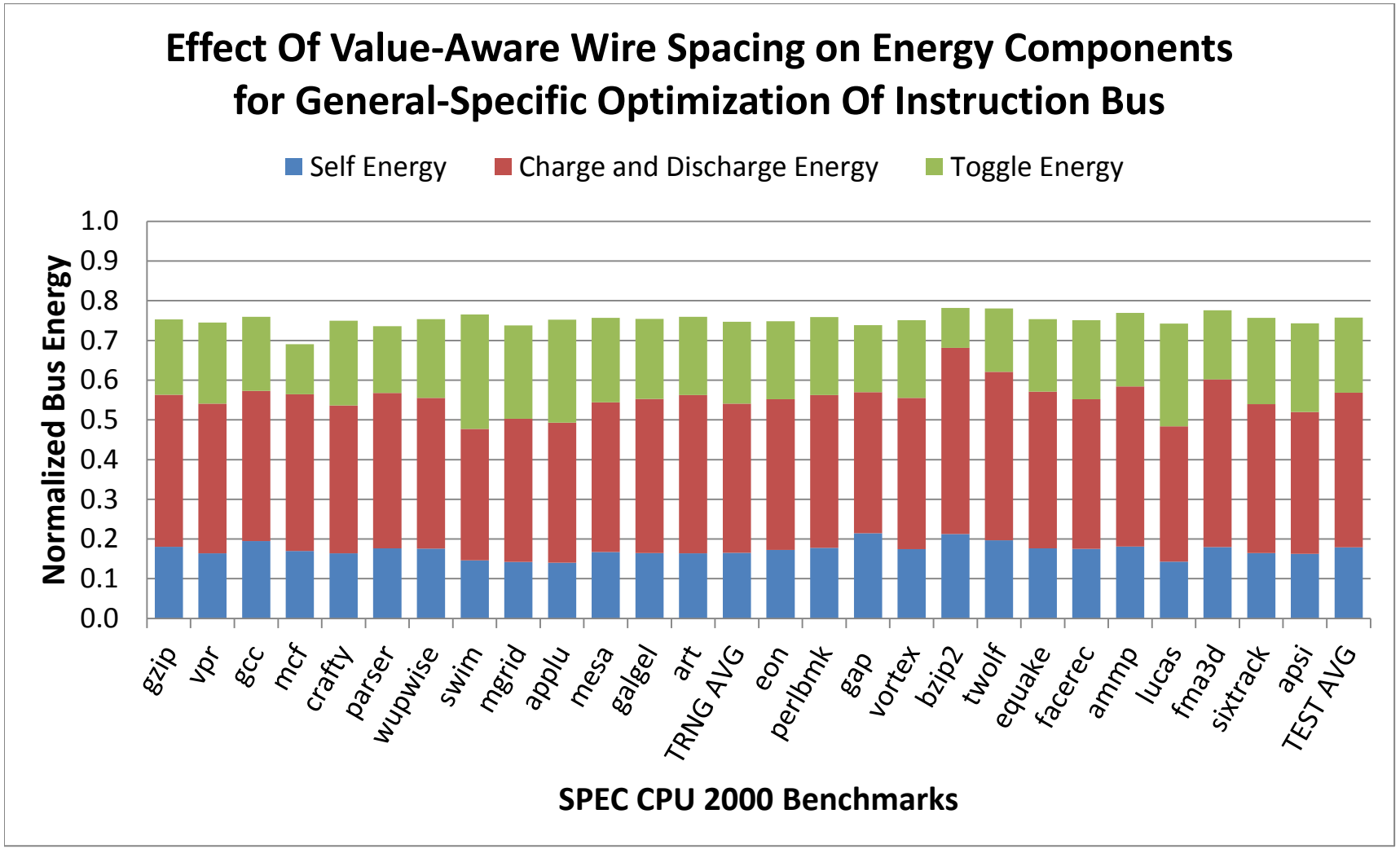


Figure 4.9: **General-Purpose Optimization for Instruction Bus:** Bus energy of SPEC CPU2K training (TRNG) and test (TEST) programs, due to value-aware wire spacing (VAWS), normalized w.r.t the original uncoded bus energy.

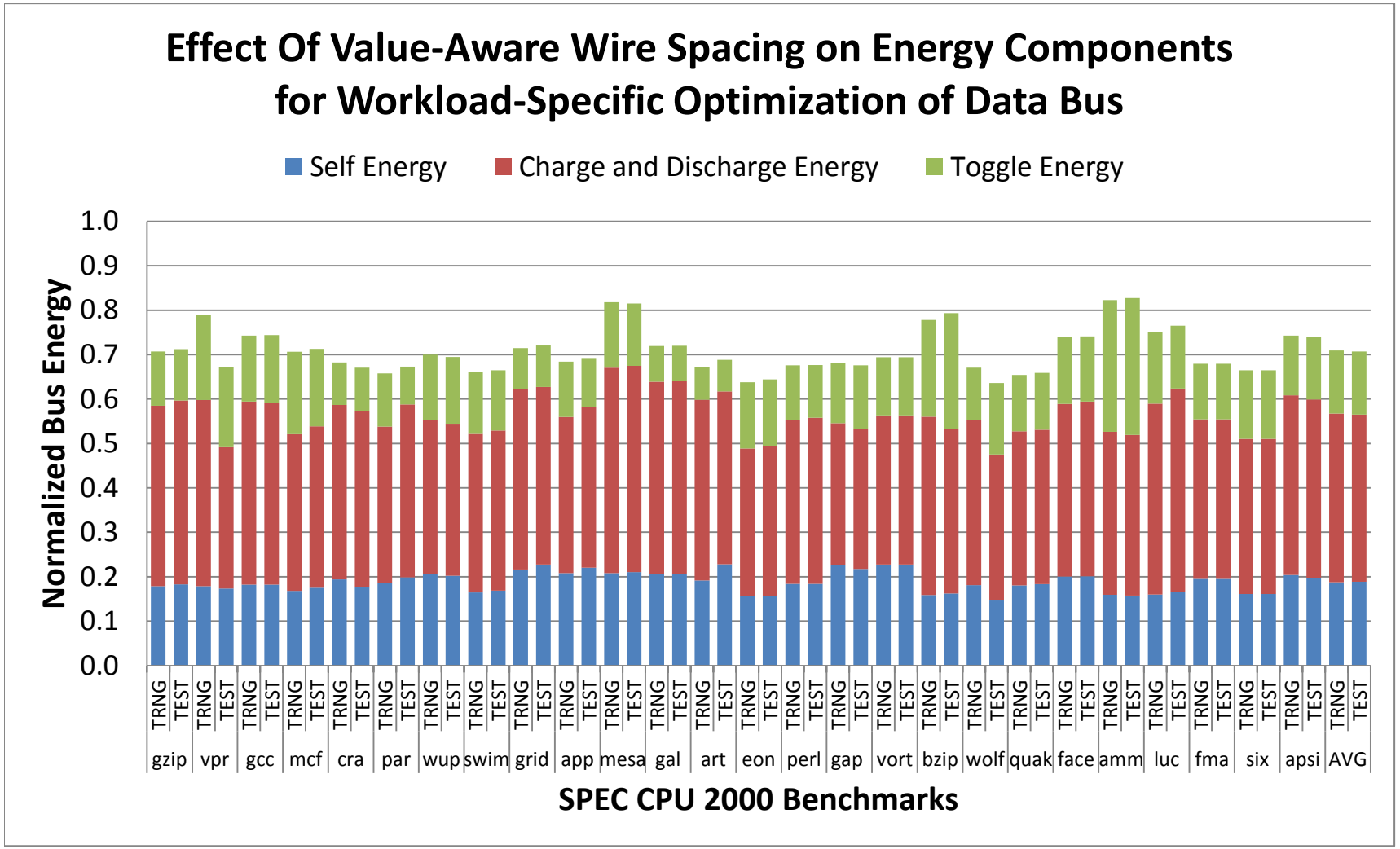


Figure 4.10: **Workload-Specific Optimization for Data Bus:** Bus energy of training (TRNG) and test (TEST) samples of SPEC2K benchmarks, due to value-aware wire spacing (VAWS), normalized w.r.t original uncoded bus energy.

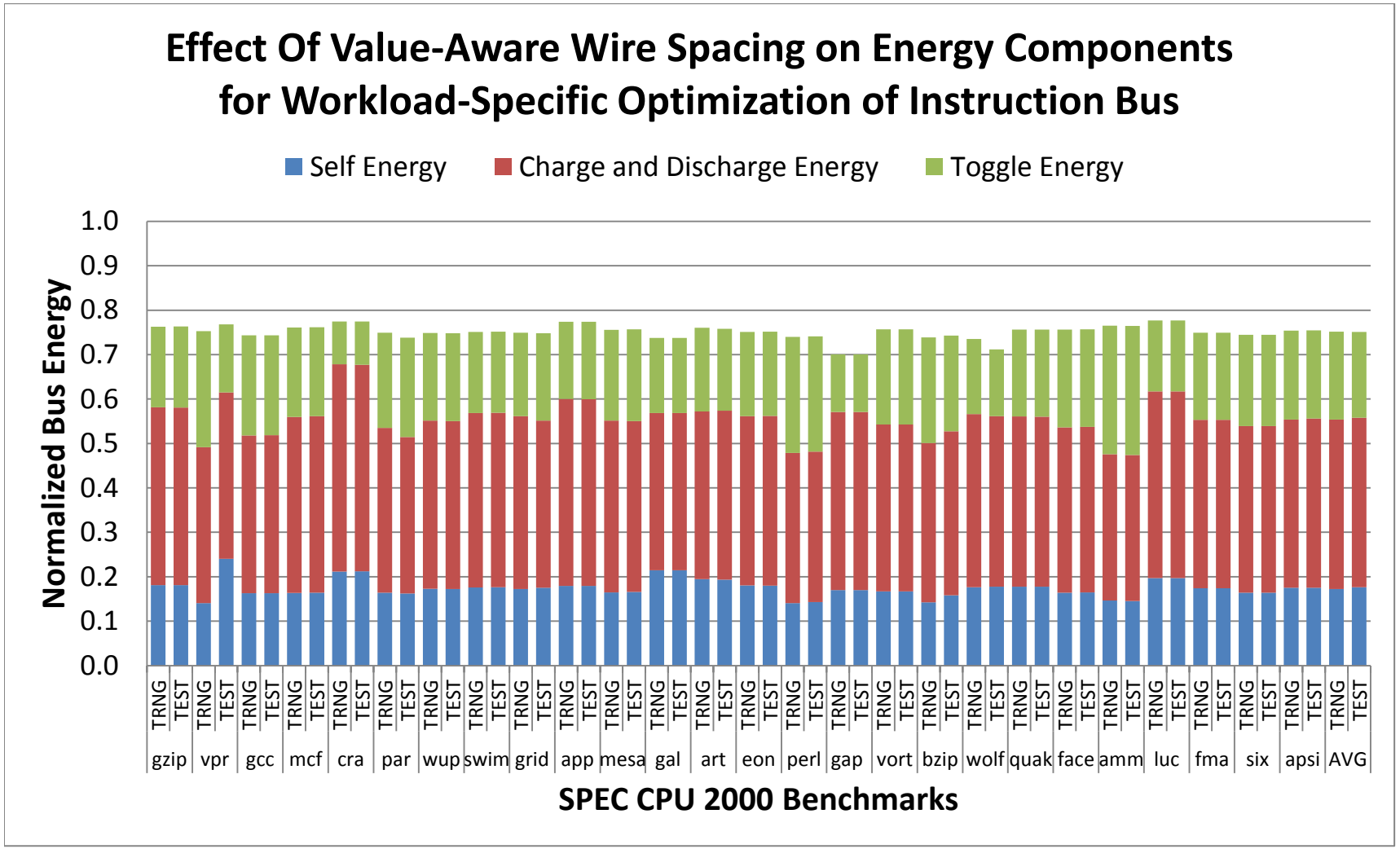


Figure 4.11: **Workload-Specific Optimization for Instruction Bus:** Bus energy of training (TRNG) and test (TEST) samples of SPEC2K benchmarks, due to value-aware wire spacing (VAWS), normalized w.r.t original uncoded bus energy.

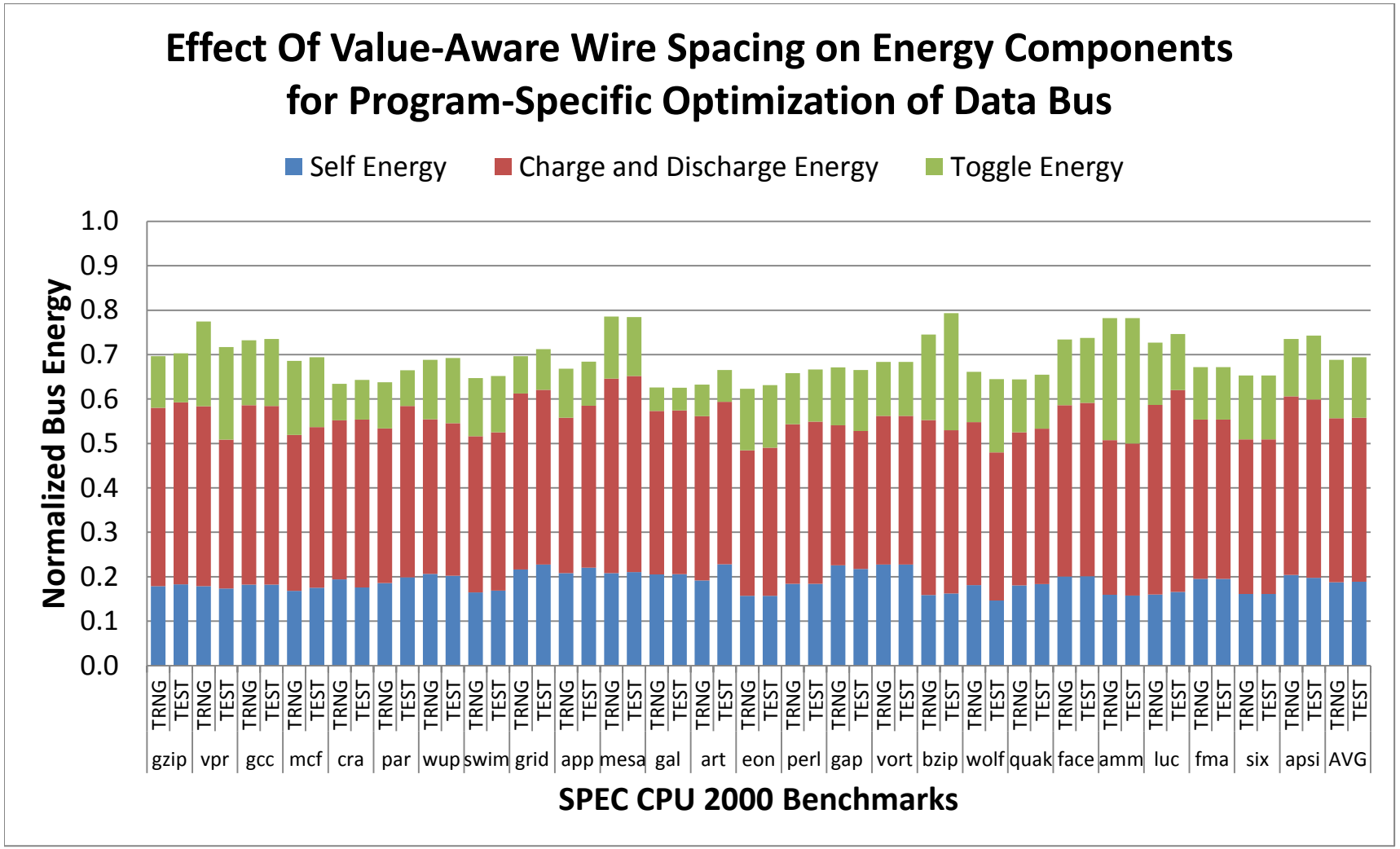


Figure 4.12: **Program-Specific Optimization for Data Bus:** Bus energy of training (TRNG) and test (TEST) samples of SPEC2K benchmarks, due to value-aware wire spacing (VAWS), normalized w.r.t original uncoded bus energy.

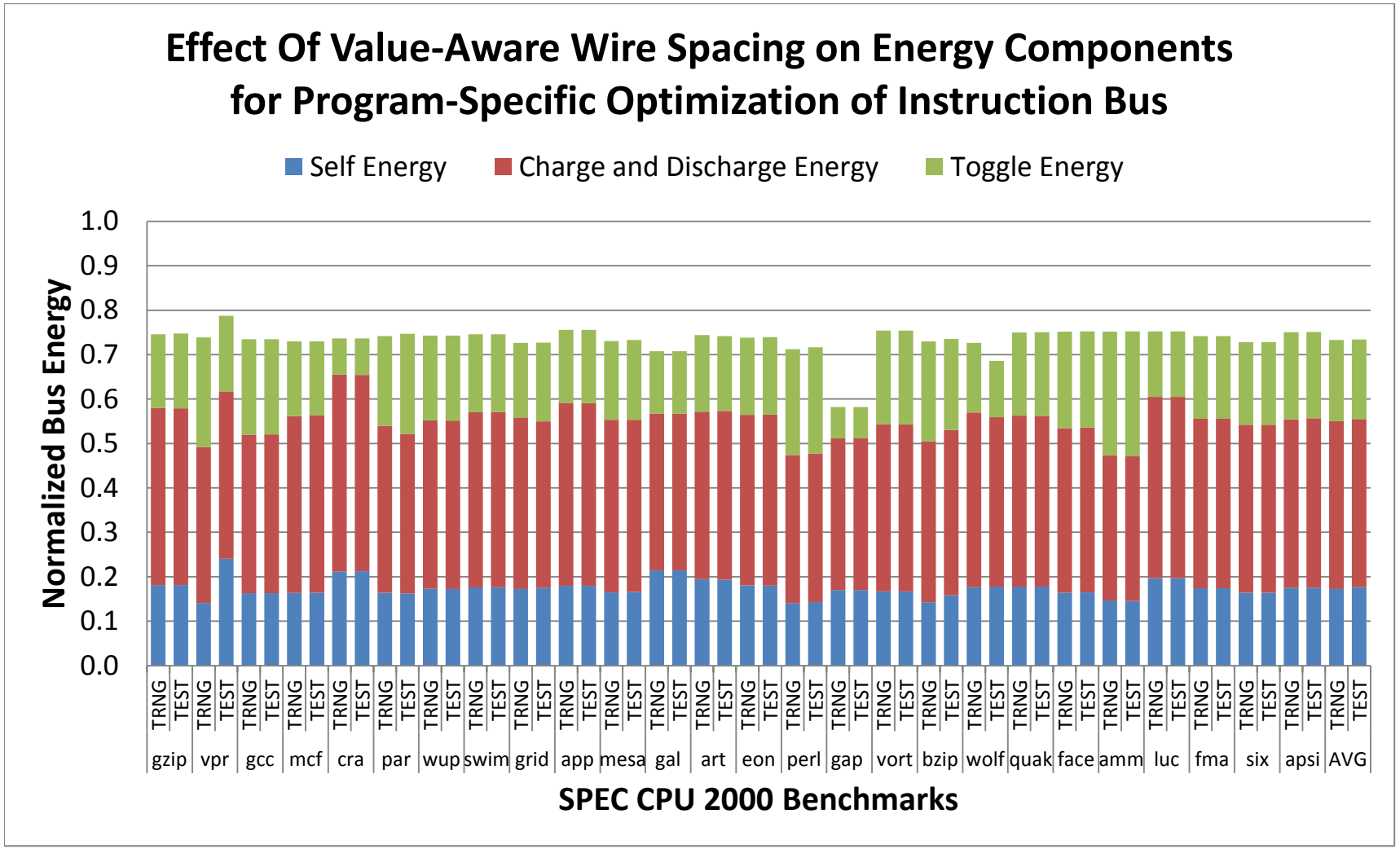
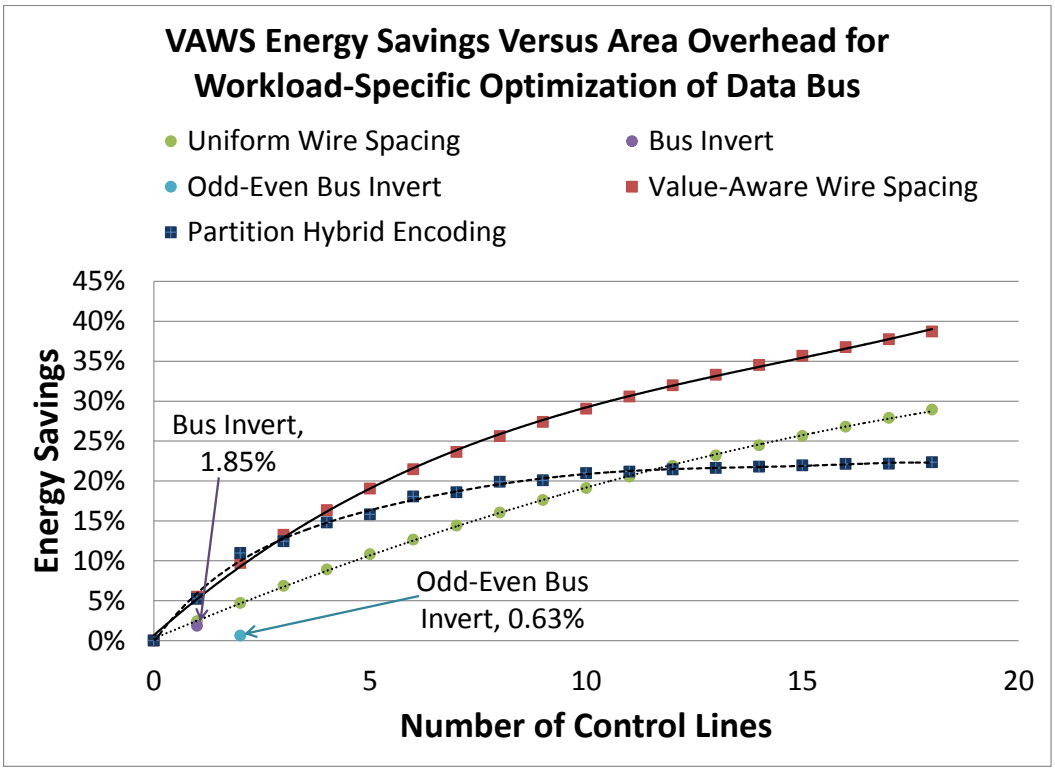


Figure 4.13: **Program-Specific Optimization for Instruction Bus:** Bus energy of training (TRNG) and test (TEST) samples of SPEC2K benchmarks, due to value-aware wire spacing (VAWS), normalized w.r.t original uncoded bus energy.



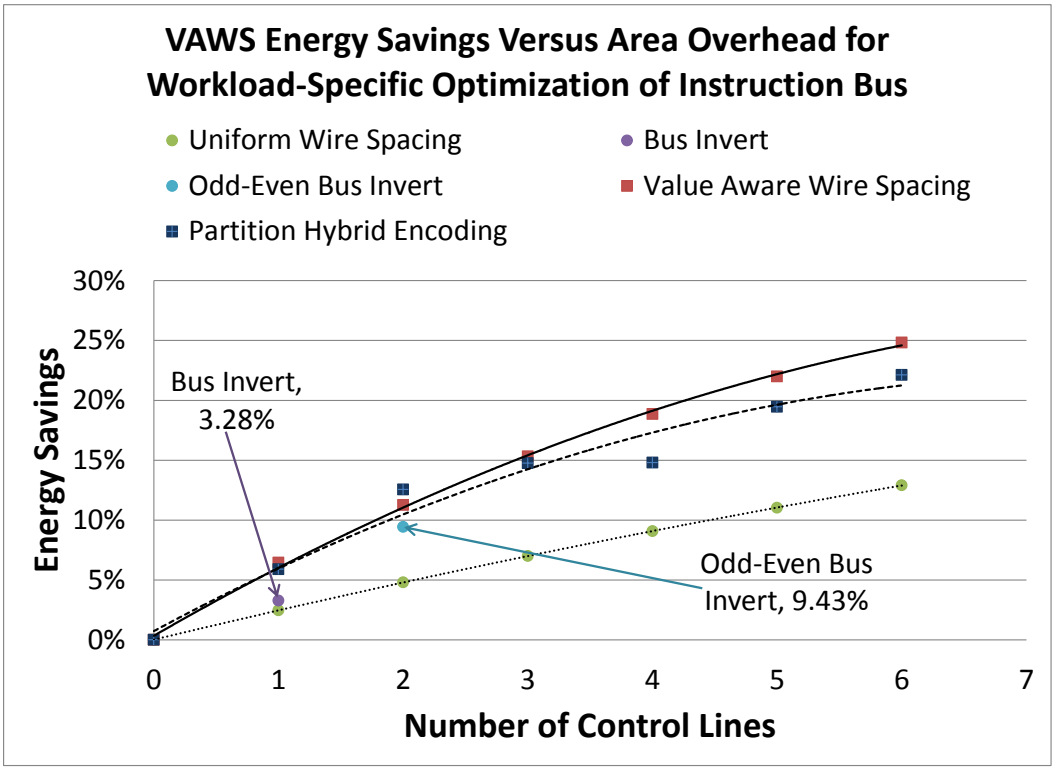


Figure 4.14: Energy savings for value-aware wire spacing (VAWS) for different area overheads compared to value-aware bus encoding technique (partitioned hybrid encoding) and uniform wire spacing for data and instruction traffic, respectively.

Chapter 5

Simultaneous Bit Ordering and Static Signaling for Multi-Objective Interconnect Optimization

5.1 Introduction

On-chip wires are a major bottleneck in the design of high-speed and low-power circuits and systems. Hence, microprocessor performance increase is only roughly proportional to square root of increase in logic complexity (Pollack's Rule) due to global interconnect delay that reduces or cancels the speed gained due to smaller transistors [1, 2]. Also, reduced interconnect dimension, increases parasitic resistance, inductance, and capacitance that aggravates interconnect delay, power consumption, and signal reliability. Thus, design of on-chip interconnects is one of the most important challenges in nanometer-scale ICs [3].

In Intel microprocessors, interconnect power accounted for approximately 51% of microprocessor power in the 130 nm technology and was projected to rise up to 65%-70% of microprocessor power, over the next few years [7]. These trends in interconnect power continues to hold even in modern architectures, such as multi-core, system-on-chip (SOC), and

network-on-chip (NOC), where interconnect energy is expected to account for almost 70% of the total chip energy [8]. This increased energy dissipation in global interconnects and use of low-k dielectric has exacerbated interconnect reliability in high performance processors due to Joule heating since wire temperature impacts wire delay and electromigration [11, 15, 16]. Rising wire temperature increases wire delays by about 5% for every 20°C rise in temperature and also degrades interconnect reliability due to electromigration [12, 13, 14].

Real world programs exhibit significant spatial, temporal and value locality that results in redundancy in instruction, address, and data bus traffics. Hence, switching (self and coupling) activities of traffic are similar across different execution regions of a program and across benchmarks. However, most encoding techniques proposed to address interconnect energy [65, 83, 82, 66, 156] are oblivious to these program characteristics and are effective for only worst-case or random traffic. Hence, these techniques are ineffective when applied to wide buses carrying correlated data [85, 164].

5.1.1 Key Contributions and Results

We present an integer linear programming (ILP) based methodology that evaluates q signaling schemes per bit and all possible bit ordering permutation for an n-bit bus to optimize interconnect design metrics, such as power, performance, and temperature. The combination of a particular way of signaling for each bits and ordering them on the bus constitutes a static encoding scheme. The ILP produces an optimal static encoding scheme by choosing exactly one signaling scheme per bit and exactly one mapping of bits to bus lines, from a total solution space of $qn \times n!$ encoding schemes, based on traffic value characteristics collected by executing real-world programs, such as SPEC benchmarks, on SimpleScalar/Alpha microarchitectural level simulator. Encoding/decoding latency, area, and energy overheads are virtually non-existent, since only one encoding scheme is supported in hardware and no control lines are needed.

Since there is substantial correlation in switching characteristics across benchmarks, our

static encoding scheme optimized for a set of programs (*training benchmarks*) works very well for a different set of programs (*test benchmarks*). This scenario, known as *general-purpose optimization*, yields an average bus dynamic energy reduction of 6.11%/7.48% for data/instruction buses. The effectiveness of the technique improves with increasing degrees of customization (suitable for particular application domains or embedded systems) to obtain average bus energy reductions of 7.49%/14.77% for workload-specific and 17.49%/26.23% for program-specific optimization scenarios for data/instruction buses. These average percentage bus energy reductions for our static encoding schemes are better than more complex dynamic encoding schemes, such as bus invert (BI) [65] and OEBI [66].

Lowering bus energy does not necessarily lower the highest temperature attained by a bus wire during program run, or peak wire temperature, because peak wire temperature depends on the spatial and temporal distribution of energy. To reduce peak temperature, we present a novel method to efficiently explore peak wire temperature and total bus dynamic energy trade-off space, using a steady-state wire temperature model [86]. Based on this temperature model, we introduce thermal constraints into our ILP framework for energy optimization that allows designer to trade-off total bus dynamic energy reduction in peak wire temperature, as desired. We demonstrate the efficacy of this approach by using data bus traffic of *lucas* benchmark and show temperature reduction of 1°C for less than 2% increase in energy compared to energy optimal static scheme. We also present a novel approach to design a static encoding scheme to reduce the number of cycles required for variable cycle bus transmission. The proposed approach reduces the number of cycles required for transmission by 21.24% and 10.90% for data and instruction buses, receptively, and can be integrated in the ILP framework to explore suitable energy-performance trade-offs.

Next, we present the different signal choices for a bit, in Section 5.2. Then, we describe our ILP framework, in Section 5.3, for finding an optimal static encoding scheme. Followed by Section 5.4, were we describes how various interconnect design metrics, such as energy, temperature, and performance, are optimized using the ILP-framework. Finally, we discuss

our results in Section 5.5.

5.2 Static Signaling Schemes

In this section, we describe the five static signaling choices considered in our optimization technique. Each bit in the bus is assigned a signaling mode from a choice of: original signaling (**org**), inverted signaling (**inv**), transition signaling (**trs**), inverted transition signaling (**itr**), and Markov signaling (**mrk**). Though, inverted and inverted transition signaling do not reduce the self activity of the wire, compared to original and transition signaling, respectively, they do influence the coupling activity between the wires and hence, are considered in our optimization.

5.2.1 Original and Inverted Signaling

Our optimization technique uses original and inverted signaling as possible choices to transmit data on a bit line. In original signaling the bit value remains unchanged and transmitted as is, while inverted signaling flips the bit value from a *zero* to a *one* and from a *one* to a *zero*. Though inverted signaling does not effect the self activity of a wire compared to original signaling, it could potentially reduce coupling energy by eliminating toggle activity.

5.2.2 Transition and Inverted-Transition Signaling

Transition and inverted transition signaling are effective in reducing self activity by converting highly active bits into a series of zero and/or ones. The reduction in self activity has significant impact on coupling activity which is dependent on self activity of two neighboring bits. Like inverted signaling, inverted transition signaling and is included because it could impact coupling energy.

5.2.3 Markov-Model Based Signaling

This signaling scheme uses a small amount of hardware at the sending and receiving end of the bits selected. This scheme maintains the k-bit values of the original data previously transmitted. These k-bits define the current state of the Markov model and is used to predict the next bit value to be transmitted. A mis-prediction is indicated by flipping the current state of the bit from a *zero* to a *one* or a *one* to a *zero*.

To determine the optimal depth of the Markov model we calculated the prediction rate for different values of k ($1 \leq k \leq 9$), for each bit, using the primary sample of the 26 SPEC benchmarks. The average prediction rate of data and instruction traffic for various Markov depth, are shown in Figure 5.1. The prediction rate of most active 8-bits was used to compute the average prediction rate. Prediction rates improved with Markov depth (k), however due to gradual improvements in prediction rate and complexity of circuitry overhead we settled for $k = 4$ and $k = 2$ for data and instruction buses, respectively.

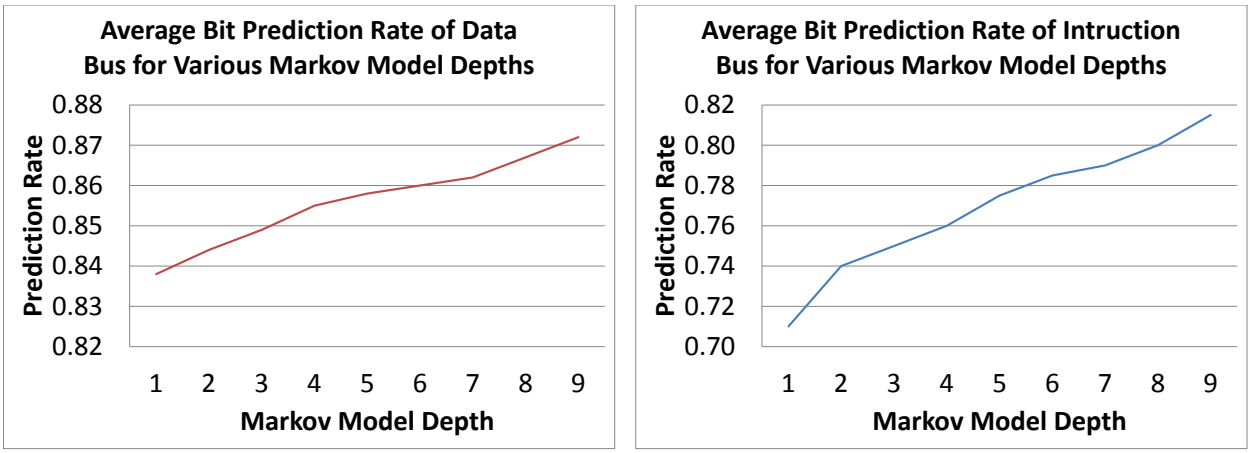


Figure 5.1: **Prediction rate versus Markov depth:** Average prediction rate computed using most active 8-bits from data and instruction buses, respectively.

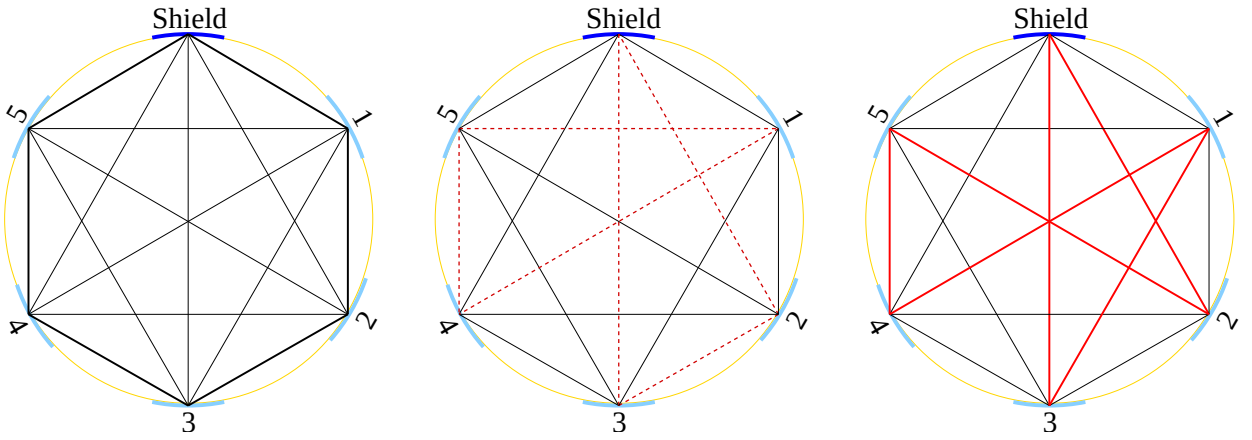
5.3 ILP Framework for Designing Value-Aware Encoding Schemes

The choice of signaling scheme influence the self and coupling activity of the bus, while bit ordering has a significant impact on coupling activity. The ILP framework, presented in this section, explores different signaling options for a bit while, simultaneously exploring all possible bit ordering of a bus, to find a static encoding scheme SBOS (simultaneous bit ordering and signaling), based on traffic characteristics which reduces the interconnect design cost. The choice of encoding scheme also impacts other interconnect design metrics, such as temperature and performance, by influencing temporal and spatial distribution of energy causing transitions. The ILP-framework provides an efficient way analyze multiple interconnect design objectives and find an optimal static encoding scheme that address all interconnect design concerns.

To simplify the understanding of our ILP framework, we, first, present the ILP algorithm for solving the minimum cost bit ordering problem which is special case of SBOS where only one signaling scheme per bit is considered. The bit ordering problem can be viewed as a tour that starts and end at the shield line, where adjacent bit (or the next city visited) is determined based on cost of the overall solution. Hence, this problem is equivalent to finding minimum cost Hamiltonian cycle. Next, we present the formulation for choosing the minimum cost signaling while maintaining the original bus ordering. Finally, we present the approach to explore all possible signaling and bit ordering combinations, simultaneously, to find the minimum cost bus configuration.

5.3.1 Minimum-Cost Bit Ordering (MCBO)

The bit ordering problem is similar to the traveling salesman problem where the order in which a salesman visits cities is determined to minimize traveling cost, given the cost of travel between any two cities. This problem of finding the minimum cost Hamiltonian cycle



(a) The shield lines at the end of the bus are merged into a single node to form a tour.

(b) Solution with two sub-tours $0 \rightarrow 2 \rightarrow 3 \rightarrow 0$ and $1 \rightarrow 4 \rightarrow 5 \rightarrow 1$ satisfying the degree constraints for each node.

(c) The minimum cost tour or Hamiltonian cycle is split at the shield node to form the reordered bus $0 \rightarrow 2 \rightarrow 5 \rightarrow 4 \rightarrow 1 \rightarrow 3 \rightarrow 0$.

Figure 5.2: Find minimum cost Hamiltonian path by solving smaller ILP's and adding constraints to eliminate subtours from previous iterations, to reduces run-time.

is NP-complete and is formulated as an ILP [96, 95]. The objective of the ILP is shown in Equation 5.1, where $c(i, j)$ is the cost of placing bits i and j adjacent to each other (also known as edge weight between i and j) and $x(i, j)$ is defined as a binary variable, to indicate if bits i and j are placed next to each other, by Equation 5.2. The number outgoing and incoming edges selected for a node is limited to 1 by Equation 5.3 and Equation 5.4, respectively, ensuring that every node is visited exactly once.

Each bit (city) is represented as a vertex in the graph $G(W, C)$, while the two shield wires, at the far end of the bus, are represented as a single vertex, as shown in Fig 5.2(a), which is the starting and end point of a tour. However, the final solution for the ILP, presented, could contain sub-tours, as shown in Figure 5.2(b). The number of sub-tour elimination constraints is exponential and leads to very large solution time. So, we solve a smaller ILP multiple times and eliminate sub-tours from previous iterations. The algorithm for finding the Hamiltonian cycle, using this iterative approach, is described by Algorithm 5.1.

Minimize

$$\sum_{\forall(i,j)\in V} \{c(i,j) \cdot x(i,j)\} \tag{5.1}$$

subject to:

$$x(i,j) \in \{0,1\},\ \forall(i,j) \in C, \tag{5.2}$$

$$\sum_{\forall j\in V} x(i,j) = 1,\ \forall i \in V, \tag{5.3}$$

$$\sum_{\forall j\in V} x(j,i) = 1,\ \forall i \in V. \tag{5.4}$$

Algorithm 5.1 Subtour from previous iterations are eliminated before ILP is re-solved, until a Hamiltonian tour is obtained.

1: Add Objective and constraints.
2: Solve ILP
3: **while** subtours found **do**
4: **for** k $\leftarrow$ 1 TO t **do**
5: $//t$ is the number of subtours.
6: $//G_k(n_k)$ is k_{th} subtours with n_k nodes.
7: Add constraint $\sum_{\forall(i,j)\in G_k} x(i,j) < n_k$
8: **end for**
9: Re-solve ILP.
10: **end while**
11: Minimum cost Hamiltonian cycle found.

5.3.2 Minimum-Cost Signaling (SIG)

While bit ordering reduces inter-wire coupling activity, the number of self transitions do not change. Static signaling schemes, which have very low circuit overhead, can be applied independently to each wire, based on traffic characteristics, to reduce self transitions in the original traffic. For example, transition signaling (trs) convert a sequence of alternating zero and one transmitted on a wire into a sequence of ones thereby reducing self transitions. On

the other hand, Markov signaling (mrk) reduce self transitions by predicting the bit to be transmitted, based on the previous k bits from the original data stream. A correct prediction of the bit value results in no activity while a mis-prediction results in a transition on the wire. Since coupling activity is dependent on the self activity of two neighboring wires, reduction in self transition activity has a significant impact on coupling activity. Hence, signaling schemes have a considerable impact on interconnect design.

We consider a set of five signaling schemes, $S = \{org, inv, trs, itr, mrk\}$, to reduce self and coupling activity. The problem of selecting a signaling scheme is formulated as an ILP, as shown in Equation 5.5, where i_p represents bit i signaled using scheme p. The original bit ordering is maintained by constraint specified in Equation 5.7 while Equation 5.8 ensures that exactly one signaling scheme is chosen for a bit.

Minimize

$$\sum_{\forall (i_p) \in V} \sum_{\forall (q) \in S} \left\{ c(i_p, (i+1)_q) \cdot x(i_p, (i+1)_q) \right\} \tag{5.5}$$

subject to:

$$x(i_p, (i+1)_q) \in \{0, 1\}, \forall (i_p, (i+1)_q) \in V, \tag{5.6}$$

$$\sum_{\forall q \ \in S} \sum_{\forall p \ \in \ S} x(i_p, (i+1)_q) = 1, \ \forall i \in V, \tag{5.7}$$

$$\sum_{\forall q \in S} x(i_p, (i+1)_q) = \sum_{\forall q \in S} x((i-1)_q, i_p), \ \forall i_p \in V. \tag{5.8}$$

5.3.3 Minimum-Cost Simultaneous Bit Ordering and Signaling (SBOS)

The choice of signaling scheme and the ordering of bits has a very significant impact on coupling activity. Hence, the problems should be solved simultaneously. In this problem, we view each bit as a super-node consisting of a set of nodes, where each node represents a signaling choice for a bit. The tour is completed by visiting each super node, exactly once. The ILP formulation for simultaneous signaling and ordering is given by Equation 5.9. The number outgoing and incoming edge selected for a super-node is limited to 1 by Equation 5.11 and Equation 5.12, respectively, ensuring that every super-node is visited, exactly once. While Equation 5.13 selects exactly one node per super-node or one signaling choice per bit. Like the bit ordering problem, sub-tours from previous iterations are eliminated and the ILP is re-solved until the minimum cost Hamiltonian cycle is obtained.

Minimize

$$\sum_{\forall (ip,jq) \in V} \left\{ c(i_p, j_q) \cdot x(i_p, j_q) \right\} \tag{5.9}$$

subject to:

$$x(i_p, j_q) \in \{0, 1\}, \forall (i_p, j_q) \in V, \tag{5.10}$$

$$\sum_{\forall jq \in V} \left\{ \sum_{\forall p \in S} x(i_p, j_q) \right\} = 1, \ \forall i \in V, \tag{5.11}$$

$$\sum_{\forall jq \in V} \left\{ \sum_{\forall p \in S} x(j_q, i_p) \right\} = 1, \ \forall i \in V, \tag{5.12}$$

$$\sum_{\forall jq \in V} \left\{ x(i_p, j_q) \right\} = \sum_{\forall jq \in V} \left\{ x(j_q, i_p) \right\}, \ \forall i_p \in V. \tag{5.13}$$

Next, we describe how a cost function are defined based on interconnect design constraints such as energy, delay, and temperature.

5.4 ILP Framework for Multi-Objective Optimization

The ILP framework provides an elegant environment for multi-objective optimization, as long as all objectives can be expressed as a linear equation. In this section, we will describe design objectives, such as energy, temperature and performance as linear equations. First, we describe a general approach to multi-objective optimization in the ILP framework. Next, we take a look at energy as a design constraint, followed by temperature optimization and finally we present interconnect performance optimization. All ILP problems were solved using ILOG CPLEX [165].

5.4.1 Multi-Objective Optimization Methodology

Let, $f(\Theta_1),\ f(\Theta_2),\ \cdots,\ f(\Theta_N)$ be the linear equations describing N design objectives, $\Theta_1, \cdots, \Theta_N$, where Θ_1 is the most important objective, Θ_2 is the second most important objective, so on and so forth. The ILP for SBOS, described in the previous section, is solved for each objective in their order of importance. Hence, objective Θ_i is solved during the i_{th} iteration with additional constraints $f(\Theta_k) < \eta_k \cdot f_{min}(\Theta_k),\ \forall k \in 1,\ 2, \cdots, i-1$, where $f_{min}(\Theta_k)$ is the minimum cost of objective Θ_k obtained in the k^{th} iteration, and $\eta_k > 1$ describes the slack or penalty the designer wishes to incur to address other design objectives. The multi-objective optimization methodology is described by Algorithm 5.2.

Next, we will formulate energy, temperature, and performance objectives as linear equation which can be incorporated into the ILP framework. It should be noted that temperature objective is incorporated a little differently from other objectives, as we are interested in minimizing peak temperature rather than average temperature. This approach is described in Section 5.4.3.

Algorithm 5.2 Multi-Objective Optimization

1: Minimize Objective Θ_1
2: Solve for minimum cost Hamiltonian Cycle
3: Minimum cost = $f_{min}(\Theta_1)$
4: Slack for objective $\Theta_1 = \eta_1$
5: **for** $i \leftarrow 2$ TO N **do**
6: Minimize Objective Θ_i
7: **for** $j \leftarrow 1$ TO $i-1$ **do**
8: Add constraints $f(\Theta_j) < \eta_j \cdot f_{min}(\Theta_j)$
9: **end for**
10: Solve for minimum cost Hamiltonian Cycle
11: Minimum cost = $f_{min}(\Theta_i)$
12: Slack for objective $\Theta_i = \eta_i$
13: **end for**

5.4.2 Energy Optimization

The choice of signaling scheme influences both dynamic self and coupling energy while the choice of neighboring bit affects the dynamic coupling energy. Hence, the weight of an edge should capture both components of dynamic energy in order to minimize interconnect dynamic energy. In order to capture the total energy in the tour, we add half the self energy of bit i signaled using scheme p to every edge connected to vertex i_p because exactly two edges on vertex i_p are selected if the vertex is included in the final solution. Hence, the minimum cost Hamiltonian cycle selected would minimize the total interconnect dynamic energy. Hence, the cost of placing bits i_p and j_q next to each other is given by Equation 5.14, where $E_C(i_p, j_q)$ is the coupling energy dissipated on wire i_p due to wire j_q and $E_S(i_p)$ is the dynamic self energy dissipated on wire i_p.

$$c(i_p, j_q) = E_C(i_p, j_q) + E_C(j_q, i_q) + \frac{E_S(i_p) + E_S(j_q)}{2} \tag{5.14}$$

Most of the interconnect leakage energy dissipated are due to buffers placed along the length of the wire to improve interconnect delay. The use of dual threshold voltage inverter can be used to reduce leakage energy by exploiting the value characteristics. Since, signaling schemes considered in this work reduce dynamic energy, reducing leakage energy

becomes an assignment problem. However, leakage energy could be easily incorporated into the optimization problem, similar to self energy, by rewriting the edge cost as shown in Equation 5.15.

$$c(i_p, j_q) = E_C(i_p, j_q) + E_C(j_q, i_q) + \frac{E_S(i_p) + E_S(j_q)}{2} + \frac{E_L(i_p) + E_L(j_q)}{2} \tag{5.15}$$

5.4.3 Peak Temperature Optimization

Algorithm 5.3 Energy minimization under peak temperature constraints.

```
1: E_org energy of the original bus.
2: T_org peak temperature of the original bus.
3: T_peak = T_org
4: p = 1
5: △T = 1°K
6: repeat
7:    //Each step peak temperature is reduced by △T.
8:    T'_peak = T_peak − △T
9:    for i ← 0 to N do
10:      for j ← 0 to N do
11:         for k ← 0 to N do
12:            Compute T(j) given i and k
13:            if T(j) > T_peak then
14:               Add constraint: x(i, j) + x(j, k) ≤ 1
15:            end if
16:         end for
17:      end for
18:   end for
19:   Solve MCBO.
20:   E^p_opt energy of the MCBO solution.
21:   T^p_peak peak temperature of the MCBO solution.
22:   T_peak = T^p_peak
23:   p = p + 1
24: until New bit ordering not found
```

Minimizing bus energy does not guarantee minimum peak bus temperature because thermal coupling is a significant phenomenon in global lines, particularly when high activity wires are placed next to low activity ones [166]. The intra-layer heat transfer or thermal coupling

between the wires is likely to cause hot-spots when two high-activity wires are placed adjacent to each other. The maximum temperature on the bus occurs at such hot-spots. Since, we are interested in minimizing peak temperature and not average temperature, it is not possible formulate temperature as an objective function in the ILP. However, peak temperature can be minimized in the ILP framework by adding temperature as constraints. Algorithm 5.3 describes the approach to minimizing peak temperature using bit ordering and can be easily extended to minimize peak temperature in SBOS.

It should be noted that thermal constraints may decrease energy savings potential of SBOS but provide designers the flexibility to trade-off energy for reducing peak wire temperature. The thermal constraints were developed using the steady state thermal model. [86].

5.4.4 Average Delay Optimization

The inter-wire coupling capacitance or crosstalk determines the worst case delay for transmitting a word. Most encoding schemes that eliminate worst-case crosstalk patterns require at least 50% more area to route control line (this might not be possible for wide, 64- and 128-bit, microprocessor buses) or transmit data over multiple-cycles which does not provide any performance benefits at the system level. However, the worst-case conditions do not occur frequently. Hence, variable cycle transmission [167], where the number of transmission cycles is determined dynamically, could improve overall system performance.

In this section, we discuss how value-aware bit ordering and signaling approach reduces the various crosstalk conditions and provides performance benefits in a variable cycle interconnect architecture. In addition, timing slack in multi-cycle transmission could be exploited to reduce energy as indicated by the RAZOR architecture [168].

The delay of transmitting a word depends on the wire with the worst-case crosstalk condition. The five crosstalk conditions based on the transitions in the victim and aggressor wires are: $1 + 0r$ (*mode-0*), $1 + 1r$ (*mode-1*), $1 + 2r$ (*mode-2*), $1 + 3r$ (*mode-3*), and $1 +$

4*r* (*mode-4*). Due to the complexity of computing the frequency of each crosstalk mode for different bit ordering schemes, we use pair-wise bit information to compute the frequency of each crosstalk mode, as described below. The notion will not use signaling information, though the method could be extended for simultaneous signaling and bit ordering.

The crosstalk mode of a wire is determined by its two neighboring wires. Hence, the number of triples, of the order $\Theta(n^3)$, determine the complexity of data collection, number of variables in the ILP, and the time complexity for solving the ILP. For very wide buses, the number of such variables could vary from a few hundred thousand to over a million. Hence, we restrict our analysis to pair wise bit information.

In the remainder of the section, we will discuss how to formulate an objective function for an ILP to reduce average delay. We will restrict our discussion to reducing the frequency of crosstalk *mode-3* and *mode-4*, so that data could be transmitted within a maximum of two cycles. First, we calculate the probability of no coupling activity $\psi_{C0}(i,j) = \frac{N_{C0}(i,j)}{L}$, coupling charge/discharge $\psi_{C1}(i,j) = \frac{N_{C1}(i,j)}{L}$, and coupling toggle $\psi_{C2}(i,j) = \frac{N_{C2}(i,j)}{L}$ from a traffic trace, where N_{C0}, N_{C1}, and N_{C2} are the number of occurrences of each coupling activity, respectively, in the given trace between bits i and j and L is the length of the trace.

$$\log(\prod_{\forall(i)\in\beta}(P(i,i+1))) = \sum_{\forall i\in\beta}\log(P(i,i+1)) = \sum_{\forall i\in\beta}\Psi(i,i+1) \tag{5.16}$$

Next, we compute the probability of no *coupling toggle* transition for a given bus order and select the ordering with the highest probability, because a coupling toggle transition has to occur for crosstalk *mode-3* and *mode-4* to occur. However, this is a pessimistic approach because a *coupling toggle* need not always result in crosstalk *mode-3* and *mode-4*. The probability of no coupling transition for a bus ordering β is the product of no coupling activity between any two adjacent pair of bits in β. By taking *log* the function can be expressed as a sum of terms, as shown in Equation 5.16, where $P(i,j) = (1 - \psi_{C2}(i,j)$ and $\Psi(i,i+1) = \log P(i,i+1)$.

Maximize

$$\sum_{\forall(i,j)\in V} \{\Psi(i,j) \cdot x(i,j)\} \tag{5.17}$$

subject to:

$$x(i,j) \in \{0,1\},\ \forall(i,j) \in C, \tag{5.18}$$

$$\sum_{\forall j\in V} x(i,j) = 1,\ \forall i \in V, \tag{5.19}$$

$$\sum_{\forall j\in V} x(j,i) = 1,\ \forall i \in V. \tag{5.20}$$

Hence, the objective for reducing average delay is given by Equation 5.17, where $x(i,j)$ indicates if a bit has been selected and so is restricted to a binary value by Equation 5.18. Equation 5.19 and Equation 5.20 ensures that a bit is selected exactly once by restricting the number of incoming edges and outgoing edges to *one*, respectively. The ILP formulation looks similar to the minimum cost formulation except that the objective maximizes the delay function. In addition, the bit ordering problem for minimum average delay can be extended to signaling and bit ordering as explained previously.

5.5 Results and Discussion

First, we show the efficacy of SBOS for addressing interconnect energy. Next, we present the effectiveness of SBOS for performance optimization. Finally, we show how multi-objective optimization can be achieved using the ILP framework to meet energy and temperature constraints.

5.5.1 Energy Optimization

Figure 5.3 and Figure 5.4 show the efficacy of signaling, bit ordering and simultaneous bit ordering in reducing interconnect energy. To compare these different schemes we performed aggregated statistics from the primary sample of all 26 benchmarks to obtain energy savings of 1.83%/9.12%, 8.77%/13.07%, 8.93%/15.03% for signaling, bit ordering, and SBOS, respectively, for data/instruction traffic.

The signaling scheme reduces the self transition activity by 3.44% and 6.53% but realizes an energy savings of 1.83% and 9.12% for data and instruction traffic, respectively. The difference in self activity and energy savings is primarily due to the impact of coupling energy dissipation which accounts for around 80% of the total dynamic interconnect energy. While instruction traffic experiences 9.66% reduction in coupling energy, data traffic experiences a reduction of only 1.45%. This shows that reduction in self activity does not necessarily guarantee reduction in coupling activity which is dependent on the self activity of two adjacent wires and analyzing value characteristics of the bus is essential to reducing interconnect dynamic energy.

Bit ordering technique performs better than signaling for both data and instruction traffic as it addresses coupling energy which dominates interconnect dynamic energy, yielding an energy savings of 8.77% and 13.07%. However, this technique does not reduce the number of self transitions that occur on the interconnect. Hence, we solve the bit ordering and signaling problem simultaneously to realize the maximum energy savings of 8.93% and 15.03% for data and instruction traffic, respectively. The SBOS energy savings for data traffic is not significantly different from bit ordering as signaling schemes, such as transition signaling, which reduce self activity are not chosen. Though Markov signaling is chosen for a few bit, these bits are mostly high order bits with very low activity. Hence, energy savings are realized primarily through bit ordering.

5.5.2 Impact of Customization

Value-aware optimization is limited by the amount of available information at design time. Knowledge of programs expected to run on a processor are important to improve the efficacy of any value aware design. Hence, we consider three increasing degrees of customization.

General-purpose optimization

In this case, the technique is optimized based on aggregate traffic value characteristics from primary sample of the 13 training benchmarks (chosen randomly) and applied to the remaining 13 test benchmarks to obtain an energy savings of 6.11% and 7.48% for data and instruction traffic, respectively. Figure 5.5 and Figure 5.6 shows the energy savings for instruction and data traffic, respectively, for both training and test benchmarks. The drop-off between training and test was particularly significant for instruction traffic where the energy savings on the training traffic was 18.08%. This difference in energy savings can be attributed to the fact that every possible instruction sequence captured by the SPEC benchmark suite cannot be represented wholly by a subset of benchmarks. On the other hand, the drop-off in data traffic was relatively small where the energy savings on the training traffic was 9.60%.

Workload-specific optimization

Here the technique was optimized based on the aggregate traffic characteristics of the primary sample of all 26 benchmarks and applied individually to the secondary sample of each benchmark to obtain an energy savings of 7.49% and 14.77% for data and instruction traffic, respectively. Figure 5.7 and Figure 5.8 shows the energy savings for instruction and data traffic, respectively, for both training and test sample of each benchmark. For instruction traffic the energy savings was very similar to energy savings of 15.03% obtained on the training sample. This similarity in training and test for instruction can be attributed to the fact that SPEC benchmarks were designed to capture all possible occurrence of instruction sequence. In addition, variation in program-specific optimization indicates that the two

samples (primary and secondary) from a single program represent different set of instruction sequences occur in most benchmarks. Moreover, the effectiveness of SBOS obtained using general-purpose or workload-specific optimization on test traffic of data bus indicate that value aware encoding can be applied reliably to data bus which carries the most random information compared to address and instruction buses.

Program-specific optimization

In this case, the technique was optimized for each of the benchmark's primary sample and applied to the same benchmark's secondary sample to obtain an average energy savings of 17.49% and 26.23% for data and instruction traffic, respectively. Figure 5.9 and Figure 5.10 show the energy savings for program-specific optimization of each individual benchmark for data and instruction bus, respectively. The significant drop in average energy savings of 35.80% for training traffic for instruction bus shows that there is significant difference in energy behavior across a program. The drop-off in energy savings is also significant for data traffic where the average energy savings on the training sample was 26.52%.

5.5.3 Influence of Activity Level on Energy Savings

1. Self activity level of each benchmark is defined by the number of self transition per transmitted word. The correlation between self activity and level and energy savings is 0.07 for data traffic and 0.31 for instruction traffic. Increase in self activity should increase the potential for energy savings. However, we see no correlation in data traffic because signaling schemes such as transition and inverted transition signaling are not selected and most of the energy savings are achieved through bit ordering. Moreover, self activity level does not account for the distribution in activity level across the bus.

2. Bit ordering reduces coupling energy by placing highly correlated bits adjacent to each other. Coupling energy reduction is achieved by eliminating toggle scenarios and converting them into same side transitions or charge discharge. So energy savings

depends on the ratio of toggle transition to same side transitions. This conversion can be achieved by inverting the value on bit or by placing two bit with similar values adjacent to each other. The correlation between ratio of toggle to same side transition versus energy savings has a stronger correlation 0.45 for data traffic compared to 0.15 for instruction traffic.

5.5.4 Average Delay Optimization

The performance optimization employs a heuristic approach to eliminate the worst case crosstalk scenarios and reduce the number of cycles required for transmission. To show the efficacy of our performance optimization, we aggregated the statistics from the primary sample of all 26 benchmarks and applied the scheme to the secondary sample of each of the benchmark. The proposed technique was able to eliminate 21.24% and 10.90% of the cycles for data and instruction traffic, respectively. The impact of performance optimization are shown in Figure 5.11 and Figure 5.12, for data and instruction traffic, respectively, and compared against both original bus configuration and energy optimized bus configuration. Since, energy optimization also tries to eliminate the occurrence of coupling toggle activity, we see some performance benefit in our energy optimized SBOS configuration compared to original bus configuration.

The performance optimization technique is particularly effective in data traffic where the energy optimal (EOPT) SBOS technique was able to eliminate only 2.79% of the cycles compared to 21.24% reduction in cycles by performance optimized (POPT) SBOS. This can be attributed to the random nature of data traffic where there is a lesser degree of correlation between any two pair of bits. This randomness in data traffic compared to instruction traffic is also the reason for higher variation in results across benchmarks.

For instruction traffic the energy optimal (EOPT) SBOS was able to reduce 9.98% of the cycles which was comparable to performance optimized (POPT) SBOS because instruction traffic, is limited by the instruction set architecture where correlation between any two pair

of bits is stronger. This correlation information is lost due to the approximation of the objective function.

5.5.5 Peak Temperature Optimization

Reduction in energy consumption has a direct impact on temperature. Hence, value-aware techniques designed for energy is expected to have a positive impact on temperature. However, interconnect design is driven by peak temperature (hot-spots) rather than average temperature. Hence, temperature reduction depends on the spatial distribution of energy in addition to energy reduction. Therefore, a trade-off exists between peak temperature and energy savings. Since temperature optimization requires actual energy dissipated on a wire, we performed temperature optimization using statistics collected for data bus during execution of *lucas*. We selected data bus because transition activity was concentrated on LSB of the bus creating hot-spots and *lucas* benchmark had energy savings similar to the average data bus energy savings. The trade-off between peak energy and temperature is shown in Figure 5.13. We notice that by sacrificing only 2% of the energy savings the designer could reduce the peak temperature by 1°. By obtaining this curve the designer could make a realistic decision about the appropriate operating point.

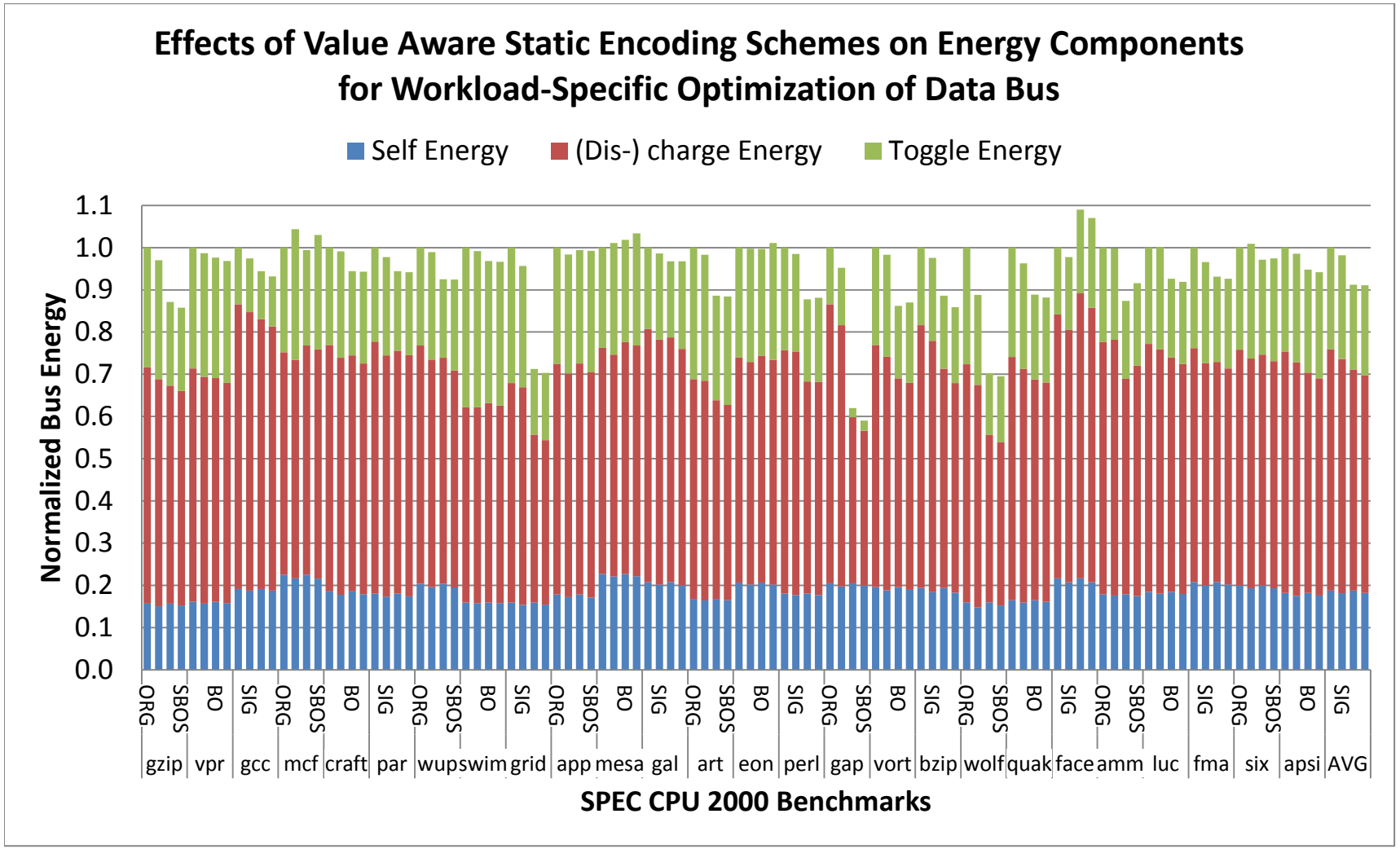


Figure 5.3: **Effect of Static Encoding Schemes on Data Bus:** Energy comparison between original uncoded bus (ORG), static signaling (SIG), bit ordering (BO), and simultaneous signaling and bit ordering (SBOS).

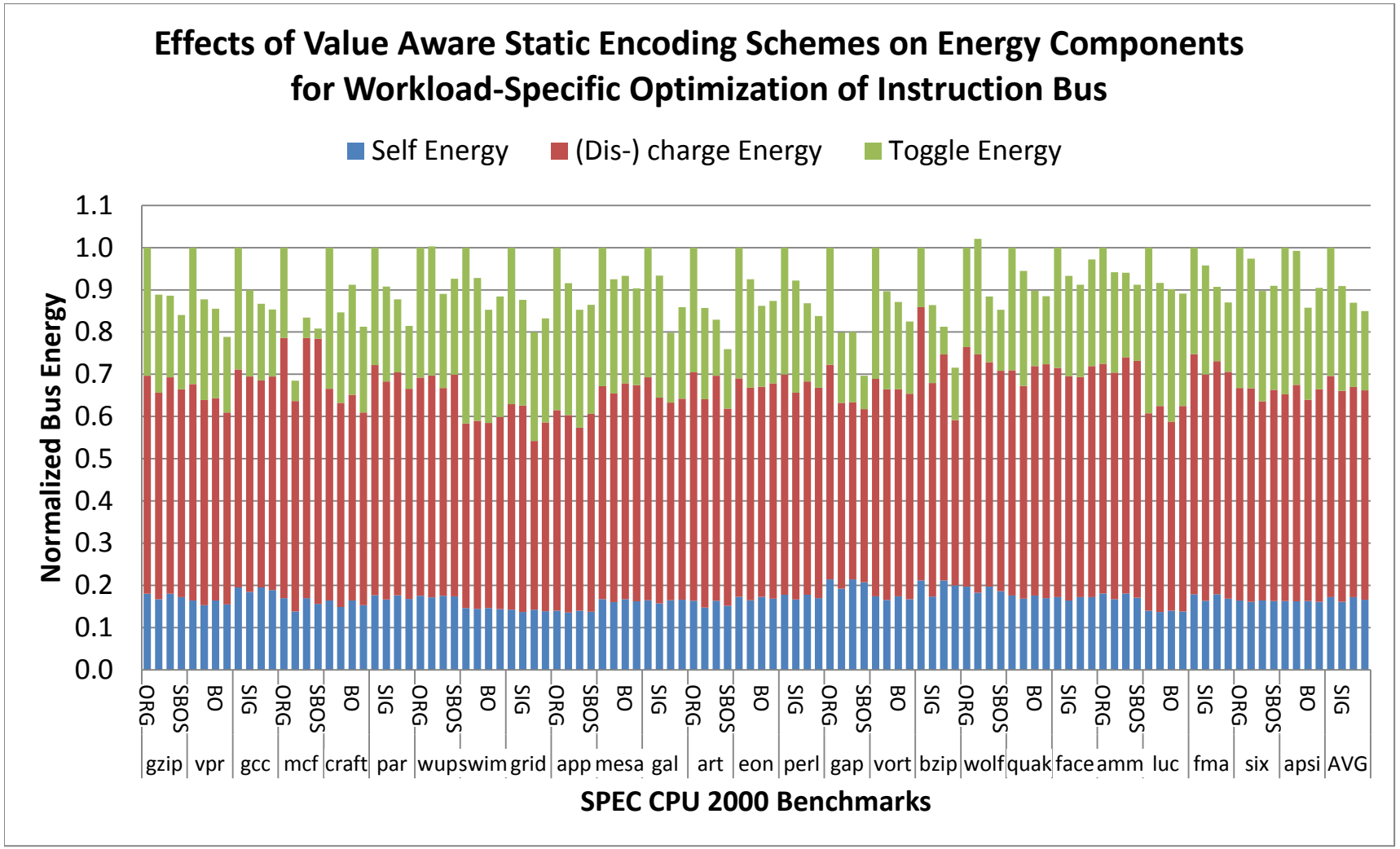


Figure 5.4: **Effect of Static Encoding Schemes on Instruction Bus:** Energy comparison between original uncoded bus (ORG), static signaling (SIG), bit ordering (BO), and simultaneous signaling and bit ordering (SBOS).

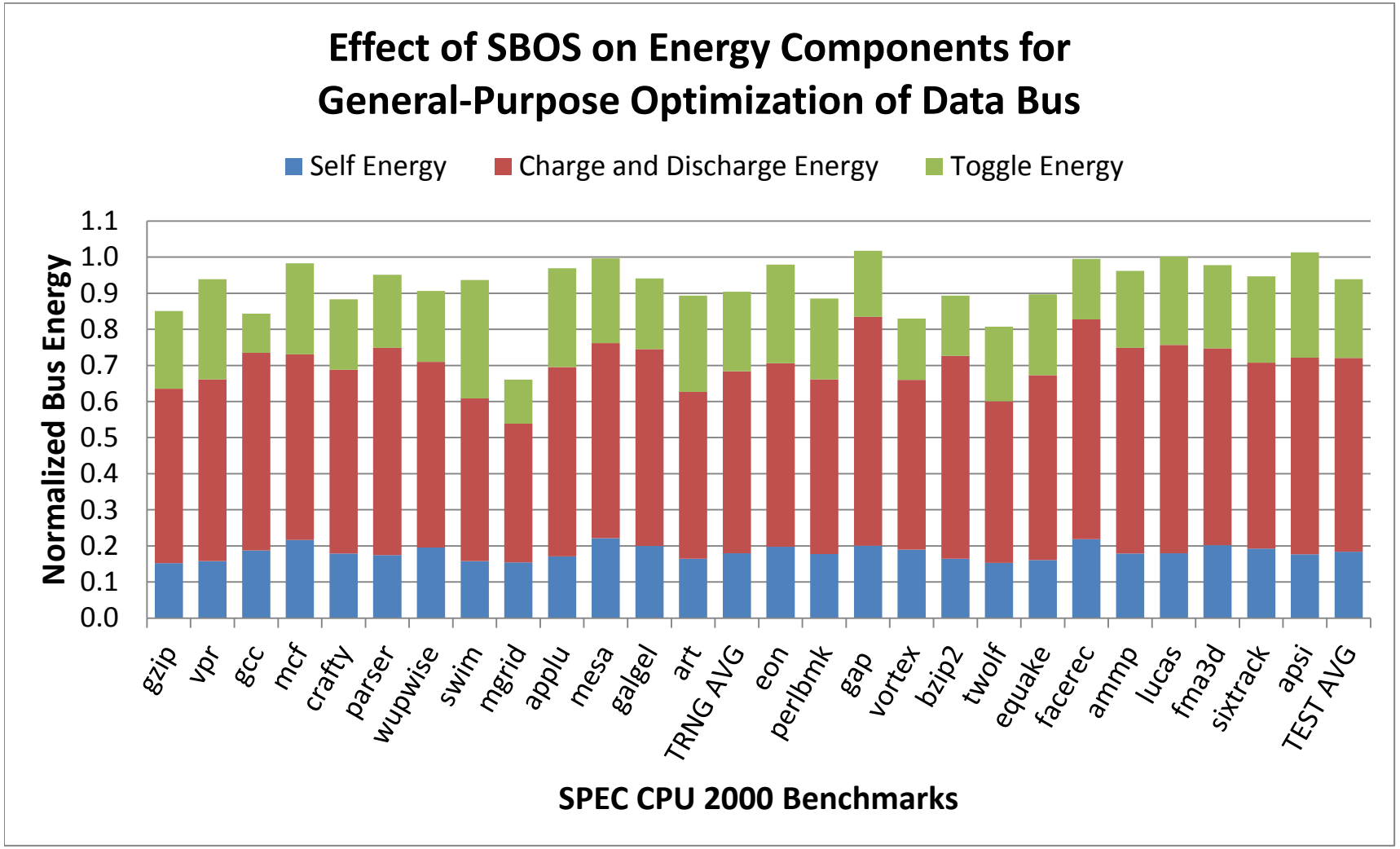


Figure 5.5: **General-Purpose Optimization of Data Traffic:** Bus energy of SPEC CPU 2000 training (TRNG) and test (TEST) programs using simultaneous bit ordering and static signaling (SBOS) scheme and normalized w.r.t the original uncoded bus energy of each program.

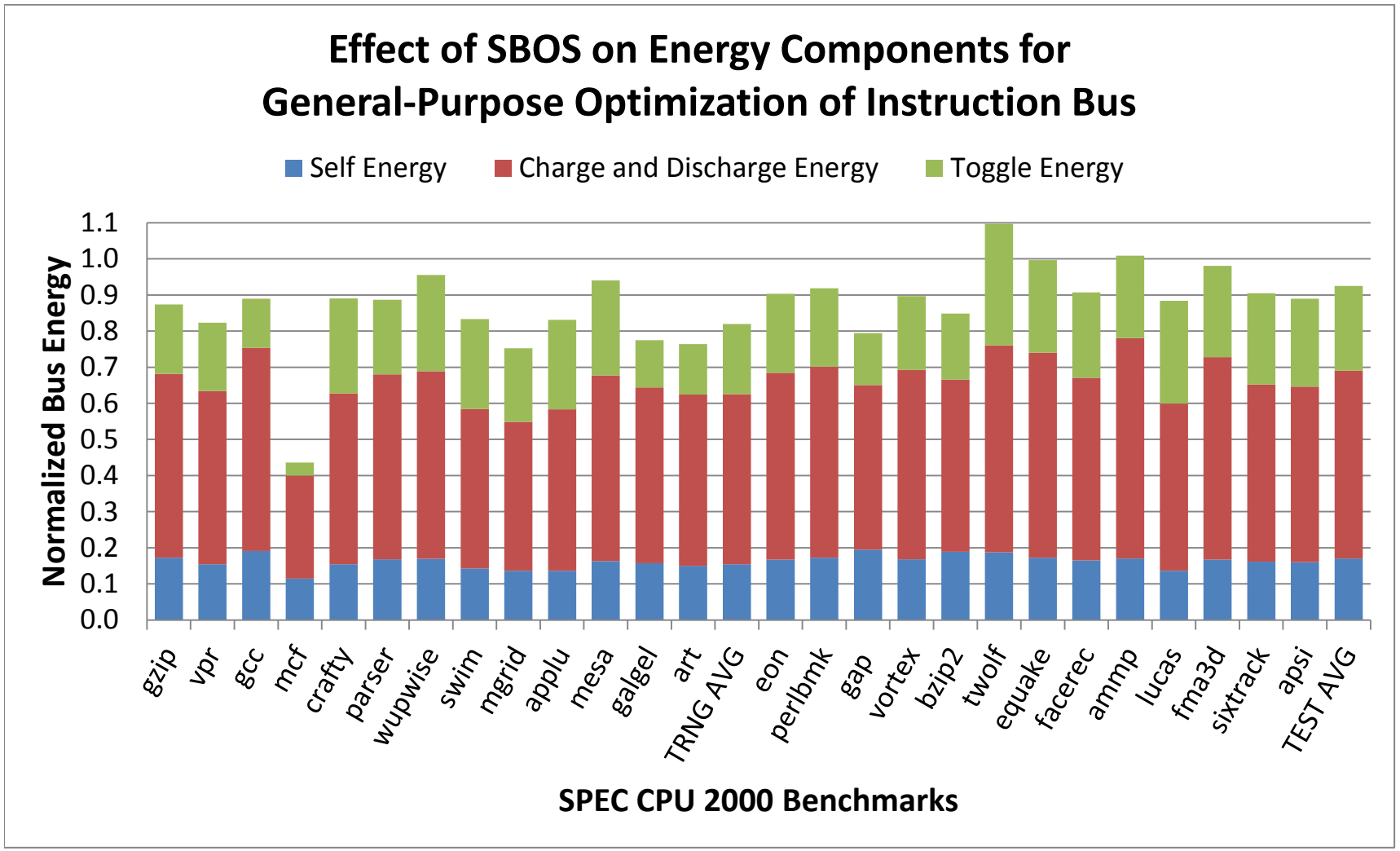


Figure 5.6: **General-Purpose Optimization of Instruction Traffic:** Bus energy of SPEC CPU 2000 training (TRNG) and test (TEST) programs using simultaneous bit ordering and static signaling (SBOS) scheme and normalized w.r.t the original uncoded bus energy of each program.

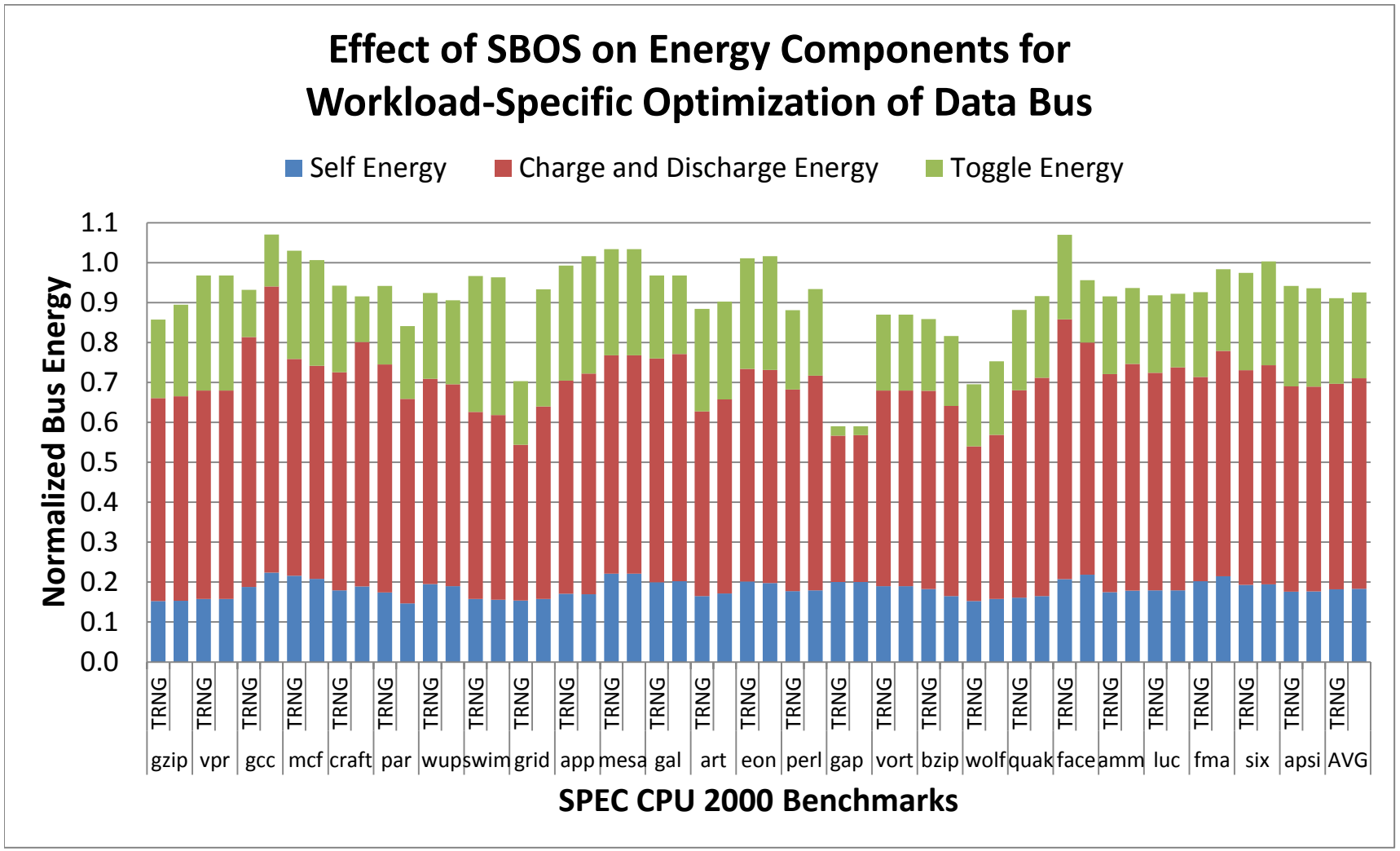


Figure 5.7: **Workload-Specific Optimization of Data Traffic:** Bus energy of training (TRNG) and test (TEST) samples of SPEC CPU2K benchmark programs, using simultaneous bit ordering and static signaling (SBOS) scheme and normalized w.r.t original uncoded bus energy of that program sample.

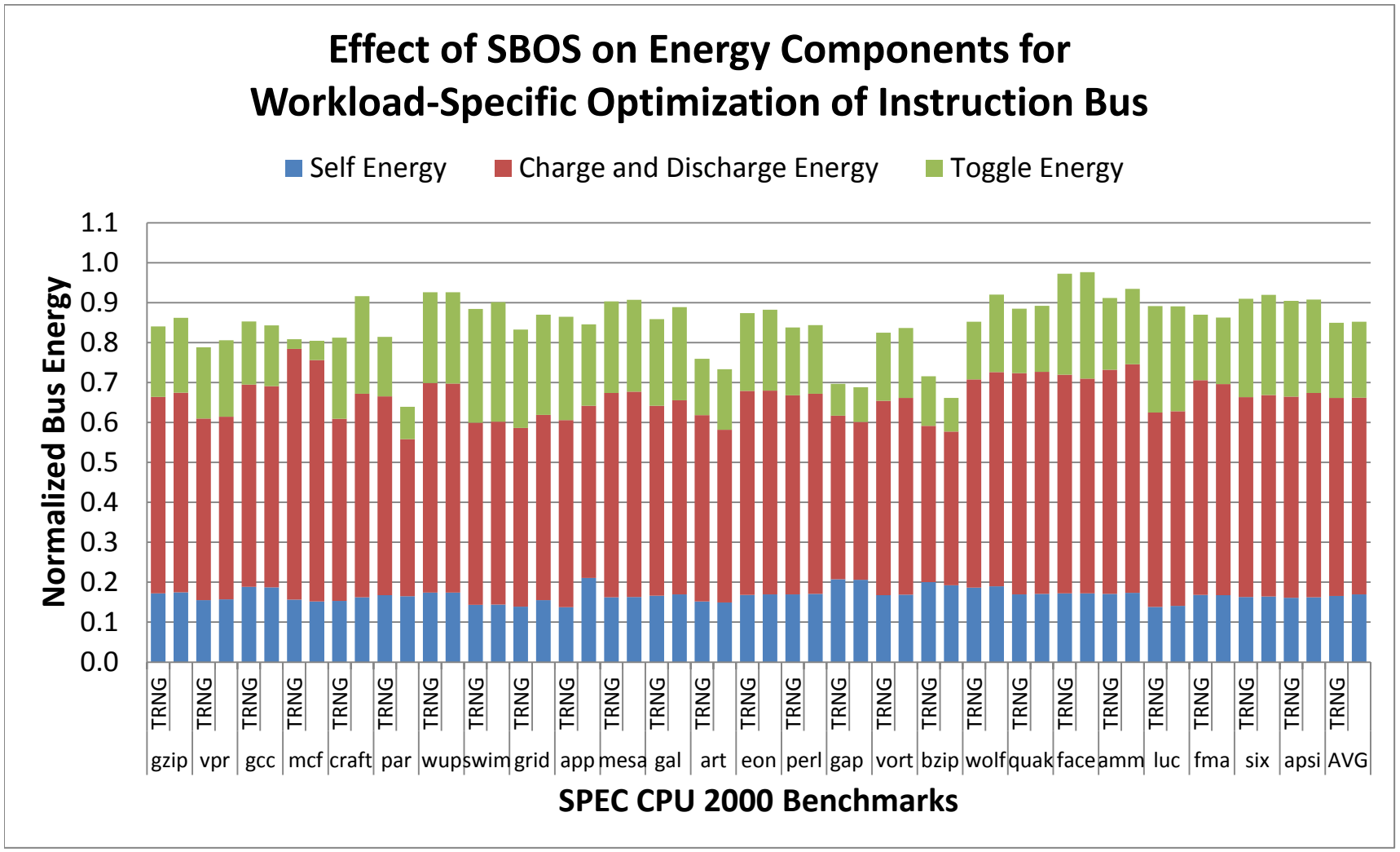


Figure 5.8: **Workload-Specific Optimization of Instruction Traffic:** Bus energy of training (TRNG) and test (TEST) samples of SPEC CPU2K benchmark programs, using simultaneous bit ordering and static signaling (SBOS) scheme and normalized w.r.t original uncoded bus energy of that program sample.

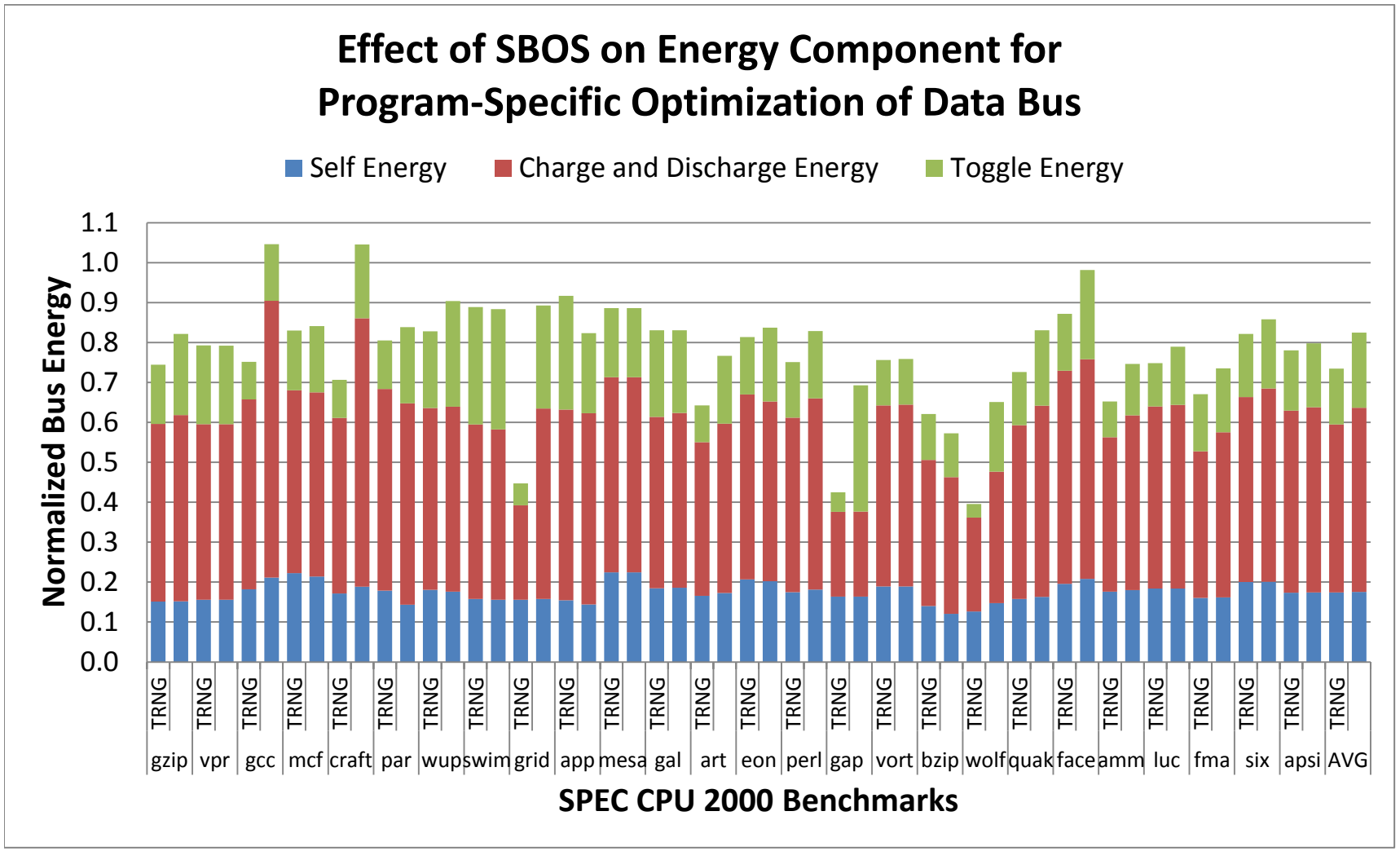


Figure 5.9: **Program-Specific Optimization of Data Traffic:** Bus energy of training (TRNG) and test (TEST) samples of SPEC CPU2K benchmark programs, using simultaneous bit ordering and static signaling (SBOS) scheme and normalized w.r.t original uncoded bus energy of that program sample.

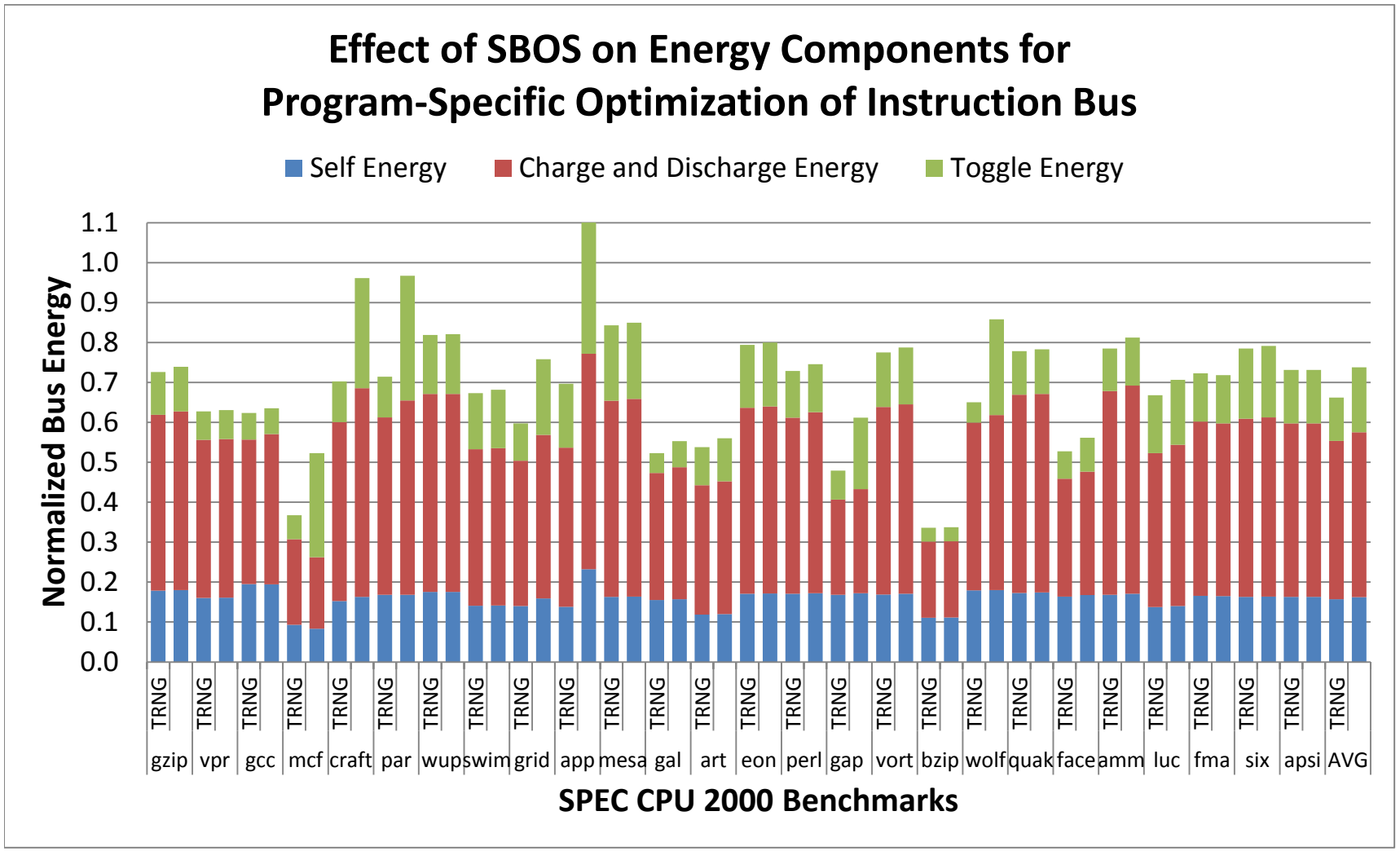


Figure 5.10: **Program-Specific Optimization of Instruction Traffic:** Bus energy of training (TRNG) and test (TEST) samples of SPEC CPU2K benchmark programs, using simultaneous bit ordering and static signaling (SBOS) scheme and normalized w.r.t original uncoded bus energy of that program sample.

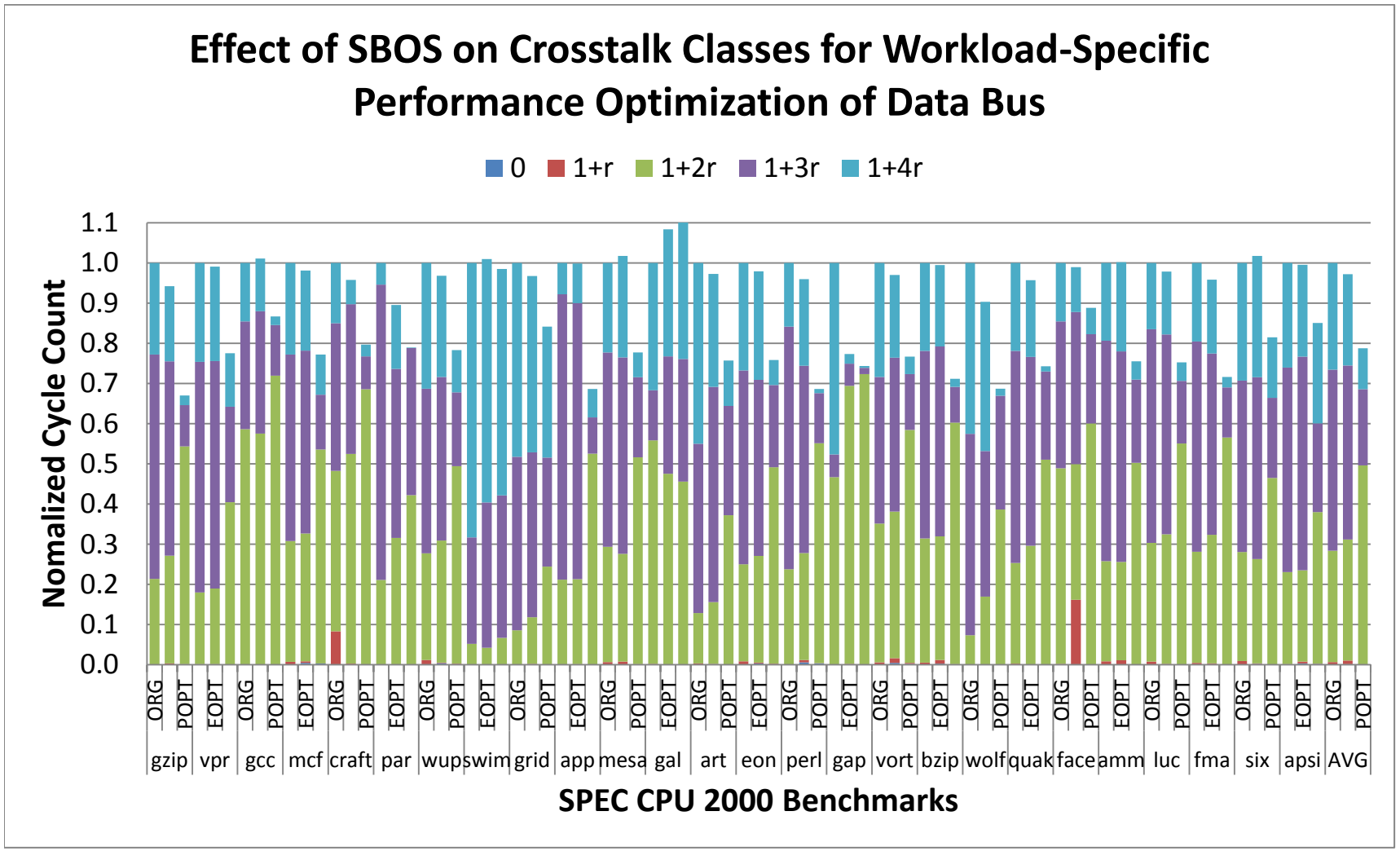


Figure 5.11: **Workload-Specific Performance Optimization of Data Traffic:** Percentage reduction in cycles with respect to original uncoded bus (ORG) using the energy optimized (EOPT) SBOS scheme and performance optimized (POPT) SBOS scheme for secondary sample of SPEC CPU 2000 benchmark programs.

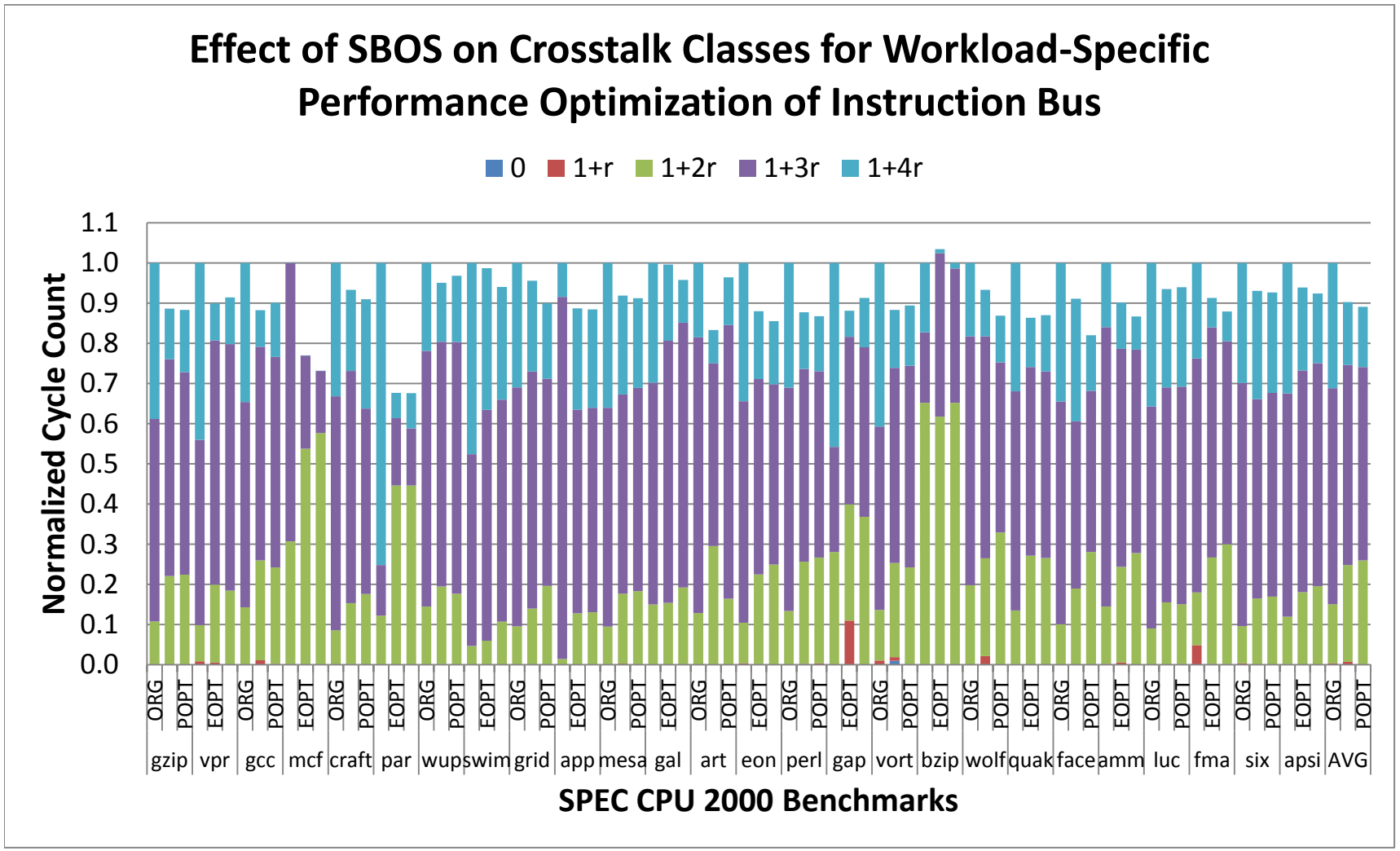


Figure 5.12: **Workload-Specific Performance Optimization of Instruction Traffic:** Percentage reduction in cycles with respect to original uncoded bus (ORG) using the energy optimized (EOPT) SBOS scheme and performance optimized (POPT) SBOS scheme for secondary sample of SPEC CPU 2000 benchmark programs.

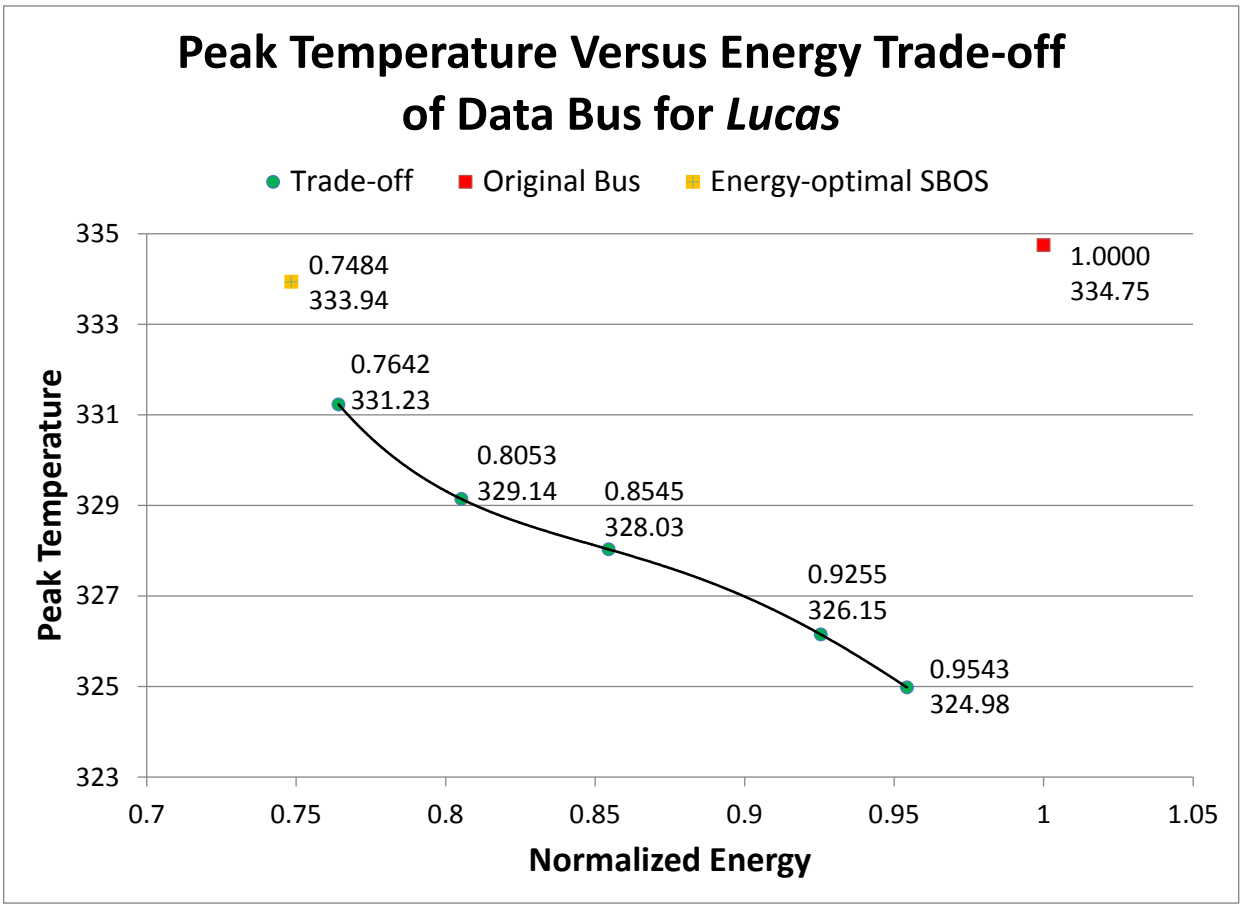


Figure 5.13: **Temperature-energy trade-off** obtained on data bus for SPEC benchmark program *lucas*.

Chapter 6

Dynamic Encoding and Bit Ordering for Multi-Objective Interconnect Design

6.1 Introduction

Reduced interconnect dimensions, due technology scaling, increases parasitic resistance, inductance, and capacitance that aggravates interconnect delay, power consumption, and signal reliability. Hence, microprocessor performance increase is only roughly proportional to square root of increase in logic complexity due to increased global interconnect delay that reduces or cancels the speed gained due to smaller transistors [1, 2]. In addition, interconnect energy particularly in long global address, instruction, and data buses, are becoming increasingly important in nanometer-scale technologies as interconnects continue to aggravate performance, power, and cost concerns.

To check growing resistance and delay of wires relative to those of transistors, wire aspect ratio of global interconnects has been gradually increased. However, this causes coupling capacitance between adjacent bus lines to become more pronounced. Solutions like introduction

of low-k dielectrics can reduce coupling capacitance to some degree but are not sufficient. Further, introduction of low-k dielectrics aggravates the problem of wire self-heating [11], which leads to increased interconnect delay [12, 13, 14] and stress related breakdowns such as electromigration (EM), and time dependent dielectric breakdown (TDDB) [11, 15, 16]. Meanwhile, interconnect power which accounted for approximately 51% of microprocessor power in the 130 nm technology [7] is expected to increase to almost 70% in modern architectures, such as multi-core, system-on-chip (SOC), and network-on-chip (NOC) [8].

Bus encoding is a widely-used approach exploiting the data-dependent nature of bus dynamic energy in which data is transmitted in an encoded form to save energy. Since real-world workloads cause bus traffic that exhibits significant spatial, temporal, and value locality, encoding schemes that exploit the characteristics of such correlated traffic can be very effective in improving bus energy efficiency. However, most existing low-power bus encoding schemes [65, 83, 82, 66, 156], which typically save energy by fully or partly inverting bus data opportunistically in selected cycles to reduce bus switching activity, are oblivious of these characteristics, and are effective only for random or worse-case (highly-changing) traffic. For realistic correlated data streams and, especially, wide (e.g., 64-bit or more) buses, these encoding schemes do not provide appreciable benefits since energy-saving encoding modes are not triggered frequently [86, 85]. While previous value-aware encoding techniques, such as partial bus invert (PBI) [88], and partitioned hybrid encoding (PHE) [85], only exploit value characteristics heuristically or are limited to a group of contiguous bits.

6.1.1 Key Contributions and Results

We present technique to design a dynamic encoding and bit ordering scheme (DEBO) with m control line and 2^m encoding modes and exactly one bit permutation. Each encoding mode inverts a particular set of bits, if that encoding mode is chosen to transmit the data in a given cycle. The encoding mode chosen in any given cycle depends on the data to be transmitted and the interconnect design metric being addressed. An integer linear programming (ILP)

framework is presented to chose exactly 2^m encoding modes and exactly one mapping of bits to bus lines, from a solution space of $2^m \times 2^n \times n!$ encoding schemes, based on traffic value characteristics. Further, to minimizing the ILP solution time, a heuristic approach that provides near optimal solution while drastically reducing run-time is also presented.

To show the efficacy of our approach, we collected data and instruction bus traffic for real-world programs, such as SPEC CPU2K benchmarks, using SimpleScalar/Alpha microarchitectural-level simulator. The dynamic encoding scheme was optimized based on aggregate traffic characteristics of each benchmark's primary sample, and tested on secondary sample of each benchmark. The dynamic encoding scheme designed for reducing interconnect energy produced an energy savings of 15.04% and 19.55% for data and instruction buses, respectively. On the other hand, dynamic encoding scheme, designed for performance, reduced the number of cycles required by 24.00% and 17.18% for data and instruction bus, respectively.

Next, in Section 6.2, we present related work. Then, describe our approach to designing simultaneous bit ordering and dynamic encoding (DEBO) scheme in Section 6.3 and discuss our results in Section 6.4.

6.2 Related Work

A number of low-power dynamic bus encoding schemes have been developed. These include schemes, such as bus invert (BI) [65], partial bus invert (PBI) [88], and narrow bus invert (NBI) [82], that seek to reduce self energy (by reducing the average value of N_S) and schemes, such as odd/even bus invert (OEBI) [66], coupling driven bus invert [83], and coupling-based bus invert [156], that attempt to reduce coupling energy (by lowering the average value of $N_{C+D} + 4 \cdot N_T$). In general, a dynamic bus encoding scheme transmits an N-bit word in any cycle using one of (at most) 2^m encoding modes on the N signal wires and indicates the chosen mode to the receiver by setting the value of m extra control wires

$\langle W_{c,m-1}, W_{c,m-2}, \ldots, W_{c,1}, W_{c,0} \rangle$. In BI, the encoding modes are: (1) ORG: the original word as is ($W_{c,0} = 0$) and (2) INV: all bits inverted ($W_{c,0} = 1$). In OEBI, apart from ORG ($W_{c,0}W_{c,1} = 00$) and INV ($W_{c,0}W_{c,1} = 11$), there are two more encoding modes: (3) EVENI: even bits (i.e., those at even-numbered positions in the word) inverted, rest unchanged ($W_{c,0}W_{c,1} = 01$) and (4) ODDI: odd bits inverted, rest unchanged ($W_{c,0}W_{c,1} = 10$).

At the beginning of a cycle, dynamic encoding schemes evaluate a bus energy metric for each encoding mode they support by comparing the current value (for both signal and control wires) on the bus with the new bit pattern to be transmitted, and then choose the mode that minimizes the metric. In general, a dynamic bus encoding scheme transmits an N-bit word in any cycle using one of (at most) 2^m encoding modes on the N signal wires and indicates the chosen mode to the receiver by setting the value of m extra control wires $\langle W_{c,m-1}, W_{c,m-2}, \ldots, W_{c,1}, W_{c,0} \rangle$. In BI, the encoding modes are: (1) ORG: the original word as is ($W_{c,0} = 0$) and (2) INV: all bits inverted ($W_{c,0} = 1$). In OEBI, apart from ORG ($W_{c,0}W_{c,1} = 00$) and INV ($W_{c,0}W_{c,1} = 11$), there are two more encoding modes: (3) EVENI: even bits (i.e., those at even-numbered positions in the word) inverted, rest unchanged ($W_{c,0}W_{c,1} = 01$) and (4) ODDI: odd bits inverted, rest unchanged ($W_{c,0}W_{c,1} = 10$).

At the beginning of a cycle, dynamic encoding schemes evaluate a bus energy metric for each encoding mode they support by comparing the current value (for both signal and control wires) on the bus with the new bit pattern to be transmitted, and then choose the mode that minimizes the metric. In BI, the metric is self activity (N_S), and so it attempts to reduce self energy by choosing the mode (ORG or INV) that will minimize N_S in a cycle. In OEBI, the metric is coupling activity ($N_{C+D} + 4N_T$), and so it attempts to reduce coupling energy. When the encoding mode used in a cycle differs from that used for the last transmitted value, the bus transition activity is altered relative to that in the original traffic, and this signifies that an energy-metric-saving mode has been triggered. This, however, does not necessarily mean bus energy is saved if the metric is different from $N_S + R \cdot (N_{C+D} + 4 \cdot N_T)$, which is true for all previous schemes [65, 88, 82, 66, 83, 156].

Thus, in a dynamic encoding scheme, there are three sequential steps processed in hardware at the beginning of every cycle before a word is transmitted: (1) generation of bit patterns for all encoding modes supported, (2) metric calculation for these bit patterns, and (3) comparison and selection of the best bit pattern based on the metric and the setting of control wire(s) appropriately. This adds to the overall bus latency: the number of logic levels required for these steps is $O(N + m)$ to $O(\log_2(N + m))$ depending upon the hardware topologies (linear or tree, respectively) used for counting and/or metric calculation (to determine N_S or N_C), and comparison (to select the best encoding option).

Among previous schemes, only PBI and PHE bus encoding schemes account for value characteristics of the data stream. PBI uses a greedy algorithm to select a subset of, possibly non-contiguous, bus lines and applies BI to it, while the other bus lines remain uncoded. However, since it uses a heuristic approach, PBI does not fully maximize the frequency of energy-metric-saving modes for BI over the entire bus, and it also does not take into account coupling energy while selecting the BI subset. In contrast, partition hybrid encoding (PHE) [85], splits the bus into disjoint partitions and the most energy-efficient encoding scheme from among a collection of schemes—we consider bus invert (BI) [65], odd/even bus invert (OEBI) [66], and not encoding as the possible options, but any other collection of schemes can be considered—is independently applied to each partition. However, PHE is limited to a set of contiguous bits. In addition, both technique do not explore possible permutation of bits and bit line which significantly impacts coupling energy. In the next section, we present our new encoding technique that addresses the above limitations.

6.3 Simultaneous Bit Ordering and Dynamic Encoding

In comparison to static encoding scheme, described in the previous chapter, the design on dynamic encoding scheme depends on grouping words which are likely to use the same encoding modes. The encoding mode chosen for each group represents a set of bits to be

inverted during transmission of that word. The symmetric difference between any two set of bits is *not null*. Choosing an encoding mode for a group of words is similar to the design of a static encoding scheme. The ILP formulation presented for the dynamic encoding scheme, choses encoding modes based on transition characteristics of each group and between groups. The dynamic encoding schemes obtained is then used to re-label data based on the encoding mode chosen. The ILP process of identifying the encoding modes is repeated until the classification error is below a pre-defined threshold.

6.3.1 Initial Classification Through Clustering

The words in a trace can initially be grouped or classified randomly. However, this could increase the number of iterations required to reach the final solution. Hence, we use k-means clustering algorithm to group different transition patterns that occur in a given cycle using Hamming distance as our metric. The centroid for each cluster is then used as encoding modes to classify the words in the trace.

Due to the size of our trace files (where each benchmark has a few million entries), we randomly selected 50,000 words from each benchmark. Though Hamming distance was the best available metric readily available, it is a poor choice, especially for wide buses carrying correlated data. For example, in 64-bit data bus where very few bits transition in a given cycle, the clustering algorithm would group two words with no common transition bits into a single cluster. To overcome this problem we weighted each entry based on the number of transitions in that word and the frequency of such occurrence in the trace. Hence, words with 2 transitioning bits would be weighted based on the frequency of words with two bit transition. The k-mean clustering algorithm was repeated 5 times or until the number of times optimal solution was found was greater than 2. During each iteration, the EM (expectation-maximization) algorithm was run 1000 time. The k-means clustering approach is described by Algorithm 6.1 uses the C clustering library [169].

In dynamic encoding schemes the energy dissipated on the control lines could be sig-

Algorithm 6.1 K-means clustering algorithm to classify data

```
1: nrows = 1,300,000        //Selected 50,000 words from each benchmark.
2: ncolumns = bus width
3: nclusters = 2^m          //2^m is the number of encoding modes.
4: npass = 1000             //Number of times EM is run.
5: freq[i]                  //frequency of i-bits transitioning in the trace.
6: for i = 1 TO nrows do
7:    weight[i] = freq[transitions(data[i])]
8:    //transitions(data[i]) returns the number of bits transitioning
9: end for
10: repeat
11:    k-mean(nclusters, nrows, ncolumns, data, weight, npass, ifound)
12:    iter++
13: until (ifound > 1) and (iter > 5)
```

nificant. Hence, assignment of labels to a group, or control line value transmitted play a significant role in the efficacy of the dynamic encoding scheme. The labeling was chosen such that the number of self and coupling transitions on the control lines were minimized in the training trace. Since, the number of permutations for mapping the label to a group (which is $2^m!$) is small for $m = 1$ or $m = 2$, the problem was solved by examining all possible options.

6.3.2 ILP Formulation

Each bit can either transmit the original uncoded value or it's inverted form in an encoding mode. Hence, number of ways to signal a bit is 2^{2^m}. For, $m = 1$, the encoding modes in the dynamic encoding scheme are $E0$ and $E1$ and the signaling choices for a bit is $S = \{00, 01, 10, 11\}$, where 00 indicates that the bit is always transmitted in the original uncoded form in both encoding modes, while 01 indicates that the bit is inverted if encoding mode $E0$ is chosen. Similarly, 11 indicates that the bit is inverted in both encoding modes, while 10 indicates that the bit is inverted when encoding mode $E1$ is chosen. Hence, the ILP for the dynamic encoding scheme is formulated as follows:

Minimize

$$\sum_{\forall c(i_p,j_q)\in V} \left\{c(i_p, j_q) \cdot x(i_p, j_q)\right\} \tag{6.1}$$

subject to:

$$x(i_p, j_q) \in \{0, 1\}, \qquad \forall (i_p, j_q) \in V, \tag{6.2}$$

$$\sum_{\forall jq\in V} \left\{ \sum_{\forall p \in S} x(i_p, j_q) \right\} = 1, \qquad \forall i \in V, \tag{6.3}$$

$$\sum_{\forall jp\in V} \left\{ \sum_{\forall p \in S} x(j_q, i_p) \right\} = 1, \qquad \forall i \in V, \tag{6.4}$$

$$\sum_{\forall jq\in V} \left\{x(i_p, j_q)\right\} = \sum_{\forall jq\in V} \left\{x(j_q, i_p)\right\}, \qquad \forall i_p \in V, \tag{6.5}$$

Equation 6.1 is the objective function of the ILP, where $c(i_p, j_q)$ represents cost of placing bit i, signaled using scheme p, and j, signaled using scheme q are place next to each other, $x(i_p, j_q)$ indicates if the edge is included in the final solution. Equation 6.2 restricts $x(i_p, j_q)$ to a binary value. The minimum cost Hamiltonian cycle is obtained by solving multiple smaller ILP's and eliminating subtours from previous iteration, as discussed in the previous chapter. The dynamic encoding scheme obtained is then applied to the training traffic and the words are relabeled based on the encoding mode chosen. The accuracy of the classification can be based on percentage of words that were re-labeled or the energy reduction obtained w.r.t previous iteration.

6.3.3 Improving ILP Run Time

For, $m = 2$, the number of signaling choices per bit was 16 which resulted in a very large ILP problem with over a million variables for a 64-bit bus. Hence, we employed heuristic

approaches to find a suitable Hamiltonian cycle that was close to optimal. The solution obtained was within 3% of the energy for the optimal solution. The proposed solution provided solutions in a matter of minutes compared to finding the optimal solution which could take a few days. The approach presented first solves the bit ordering problem before finding the signaling scheme for each bit. The bit ordering problem is solved using a cost matrix C, where $C(i,j) = \min_{\forall p, \forall q}(c(i_p, j_q))$. Once, the bit ordering problem is solved, the signaling problem is solved.

Algorithm 6.2 Heuristic approach to find a near-optimal Hamiltonian path.

1: **for all** C(i,j) **do**
2: $C(i,j) = \min_{\forall p, \forall q}(c(i_p, j_q))$
3: **end for**
4: //MCBO is minimum cost bit ordering problem discussed in previous section.
5: //border is bit ordering obtained.
6: border = **SolveMCBO(**C**, bus_width)**
7: //MCS is minimum cost bit signaling problem discussed in previous section.
8: Sig = **SolveMCS(c, border, bus_width)**

6.4 Results and Discussion

6.4.1 Energy Optimized Dynamic Encoding Scheme

In this section, we present results for our bus dynamic encoding scheme for *two* control lines, accounting for energy dissipated on the additional bus lines. We designed out dynamic encoding scheme with *two* control lines because odd-even bus invert OEBI [66], an existing encoding technique, employs two control lines. The dynamic encoding scheme was based aggregate traffic characteristics of the primary sample of all 26 SPEC benchmarks and its efficacy tested by applying the scheme to the secondary sample of all 26 benchmarks. The cost function for energy dissipated between two bits as discussed in the previous chapter. The dynamic encoding scheme optimized using this approach, also know as workload-specific optimization, is compared against BI, OEBI, and PHE. All energy results are in the form

of *normalized bus energy*, where energy components for each benchmark is normalized w.r.t. the energy dissipated on the original uncoded bus for that benchmark.

Across the benchmarks, we found the average energy savings for BI to be only 1.85% and 3.28% for the data and instruction buses, respectively. The reason for these meager savings is the relatively small number of self transitions per word transmitted in the original uncoded bus. In comparison, average energy savings for OEBI were 0.63% and 10.45% for data and instruction buses, respectively. We can see that OEBI works effectively for 32-bit instruction bus while preforming poorly on wider 64-bit data bus. It should also be noted that BI uses only one control line while OEBI uses two control lines. Meanwhile, for PHE, a value-aware encoding scheme, the energy savings for two control lines were 10.96% and 12.56%. However, the dynamic encoding scheme obtained through the ILP approach yield an energy savings of 15.04% and 19.55%, which is 2-8 times better than BI and OEBI, and 1.5-2 times better than PHE. The energy savings for our dynamic encoding scheme across benchmarks is presented in Figure 6.1 and Figure 6.2 for data and instruction traffic, respectively. It should also be noted that our dynamic encoding scheme performs 1.5-2 times better than the static encoding scheme presented in the previous chapter. In addition, compared to static encoding scheme the energy savings is positive for each benchmark as additional encoding modes allows a dynamic scheme to adapt better to changing traffic characteristics.

6.4.2 Average Delay Optimized Dynamic Encoding Scheme

The performance optimization employs a heuristic approach to eliminate the worst-case crosstalk scenarios and reduce the number of cycles required for transmission. The cost function for performance optimization was discussed in the previous chapter. To show the efficacy of our dynamic encoding scheme for performance optimization, we aggregated the statistics from the primary sample of all 26 benchmarks and applied the scheme to the secondary sample of each of the benchmark. The proposed technique was able to eliminate 16.81% and 24.00% of the cycles for data and instruction traffic, respectively. The impact of

performance optimization on individual benchmarks are shown in Figure 6.3 and Figure 6.4, for data and instruction traffic, respectively.

While dynamic encoding works significantly better than static encoding scheme in reducing the number of worst-case crosstalk cycles for instruction traffic, the dynamic encoding scheme for data traffic was able to eliminate 16.81% of the cycles compare to static encoding scheme which was able to eliminate 21.24% of the cycles. This is because the set of bits inverted in a given cycle is restricted by the number of encoding modes. However, unlike static scheme dynamic encoding scheme is able to adapt to changing traffic characteristics and performs well across benchmarks for both data and instruction bus. In comparison, static encoding scheme has a negative impact on some benchmarks.

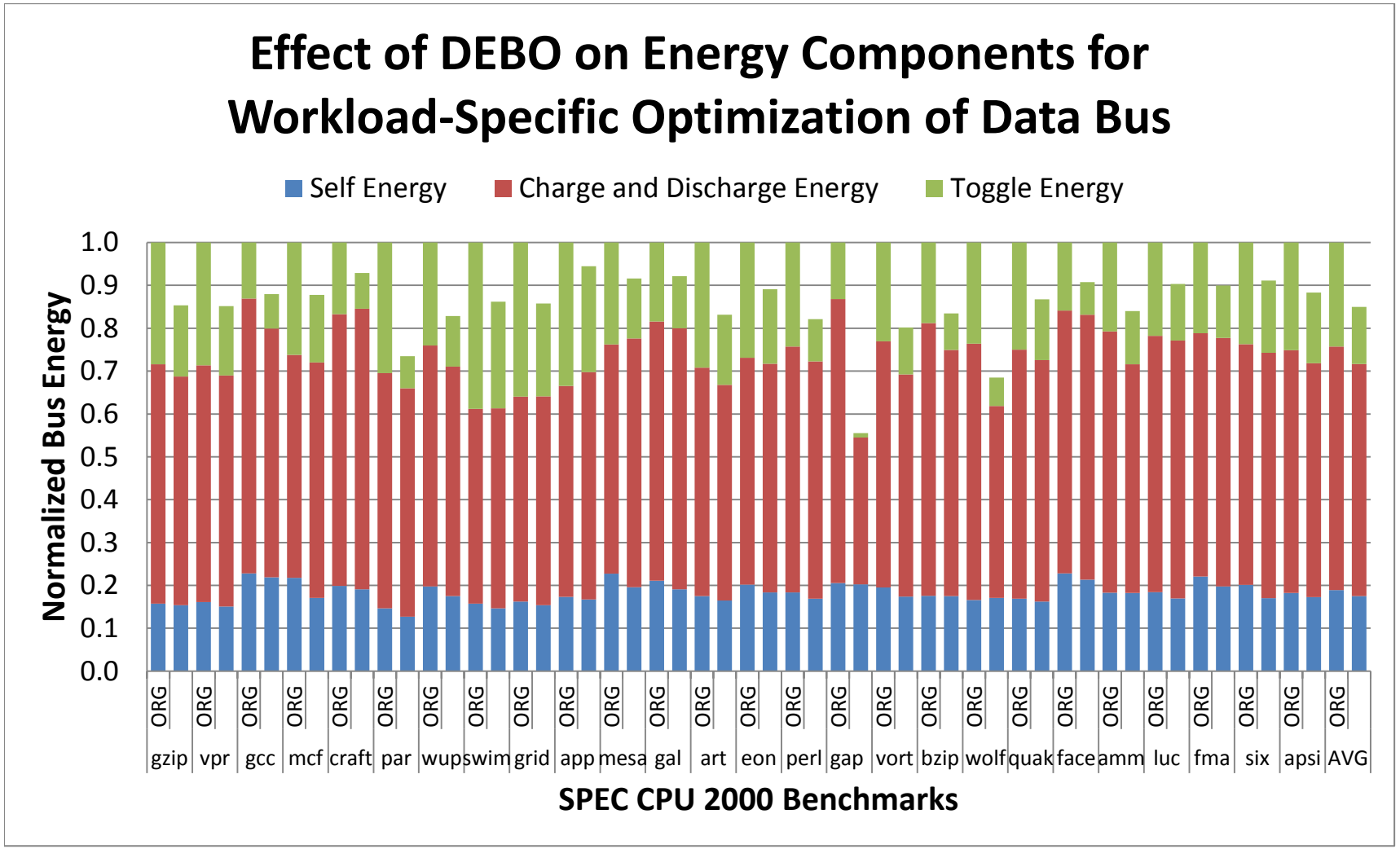


Figure 6.1: **Workload-Specific Optimization of Data Traffic:** Bus energy of SPEC CPU2K benchmark programs, using simultaneous bit ordering and dynamic encoding (DEBO) scheme and w.r.t original uncoded bus energy of that program sample.

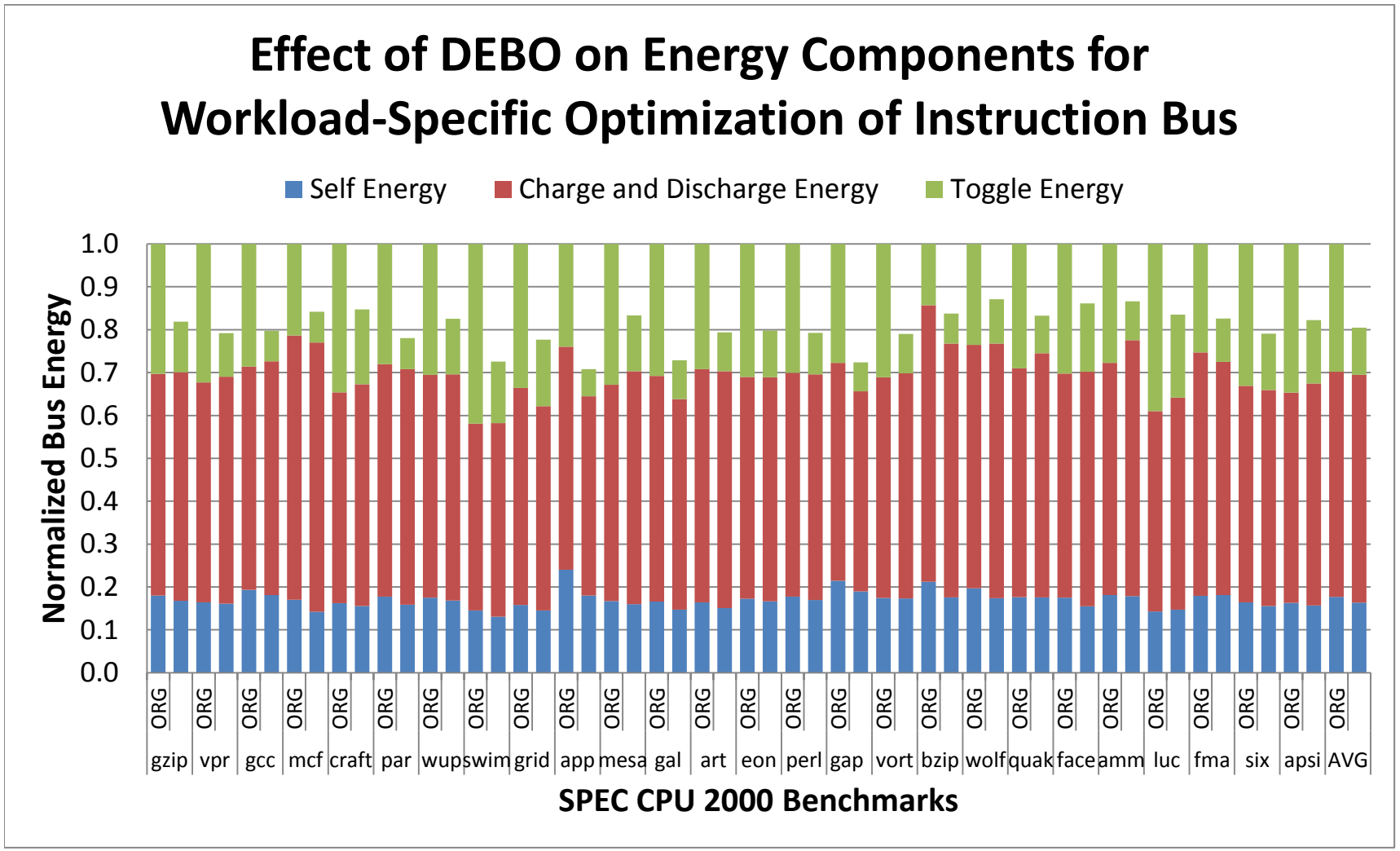


Figure 6.2: **Workload-Specific Optimization of Instruction Traffic:** Bus energy of SPEC CPU2K benchmark programs, using simultaneous bit ordering and dynamic encoding (DEBO) scheme and w.r.t original uncoded bus energy of that program sample.

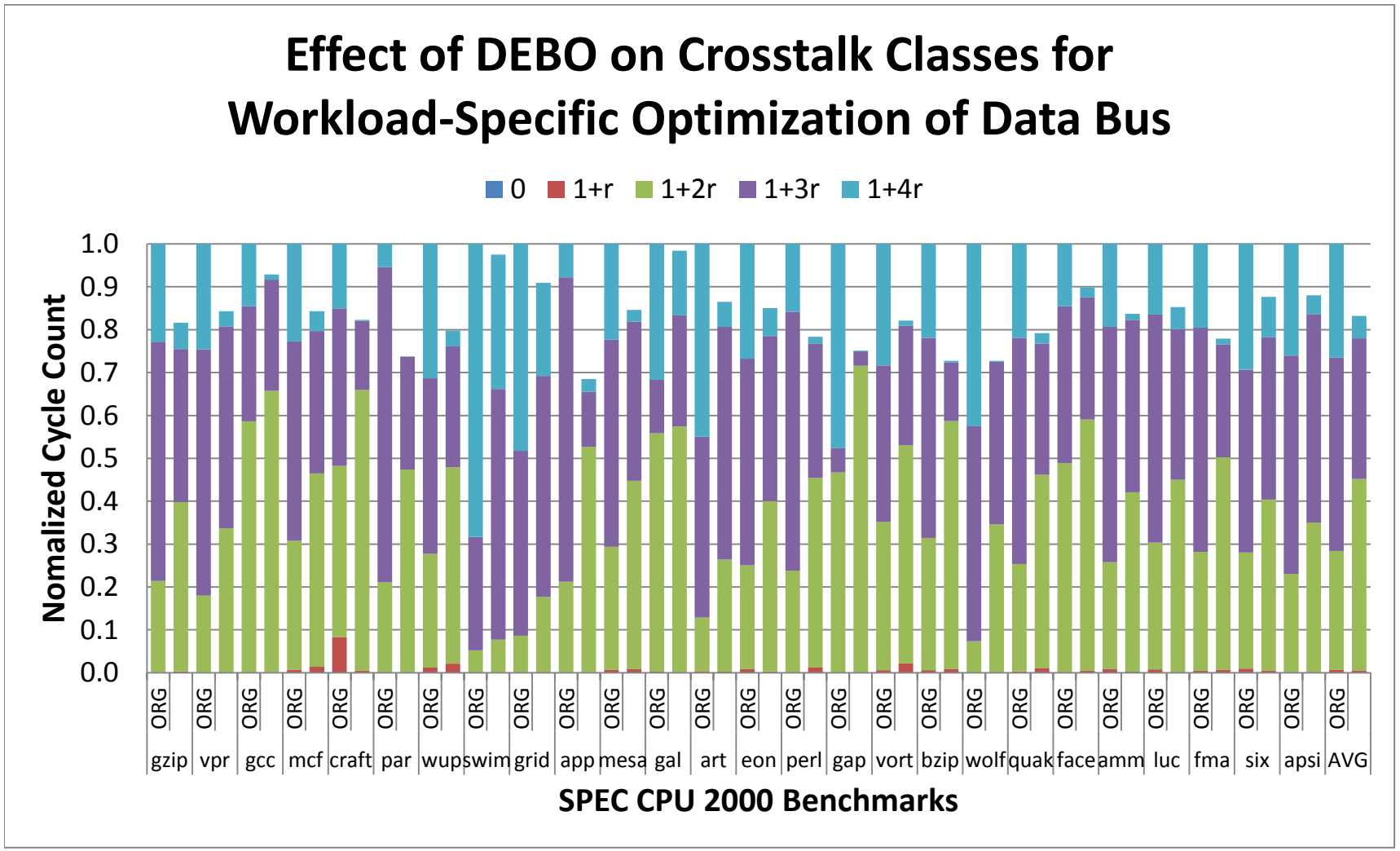


Figure 6.3: **Workload-Specific Performance Optimization of Data Traffic:** Percentage reduction in cycles with respect to original uncoded bus (ORG) using simultaneous bit ordering and dynamic encoding (DEBO) for secondary sample of SPEC CPU 2000 benchmark programs.

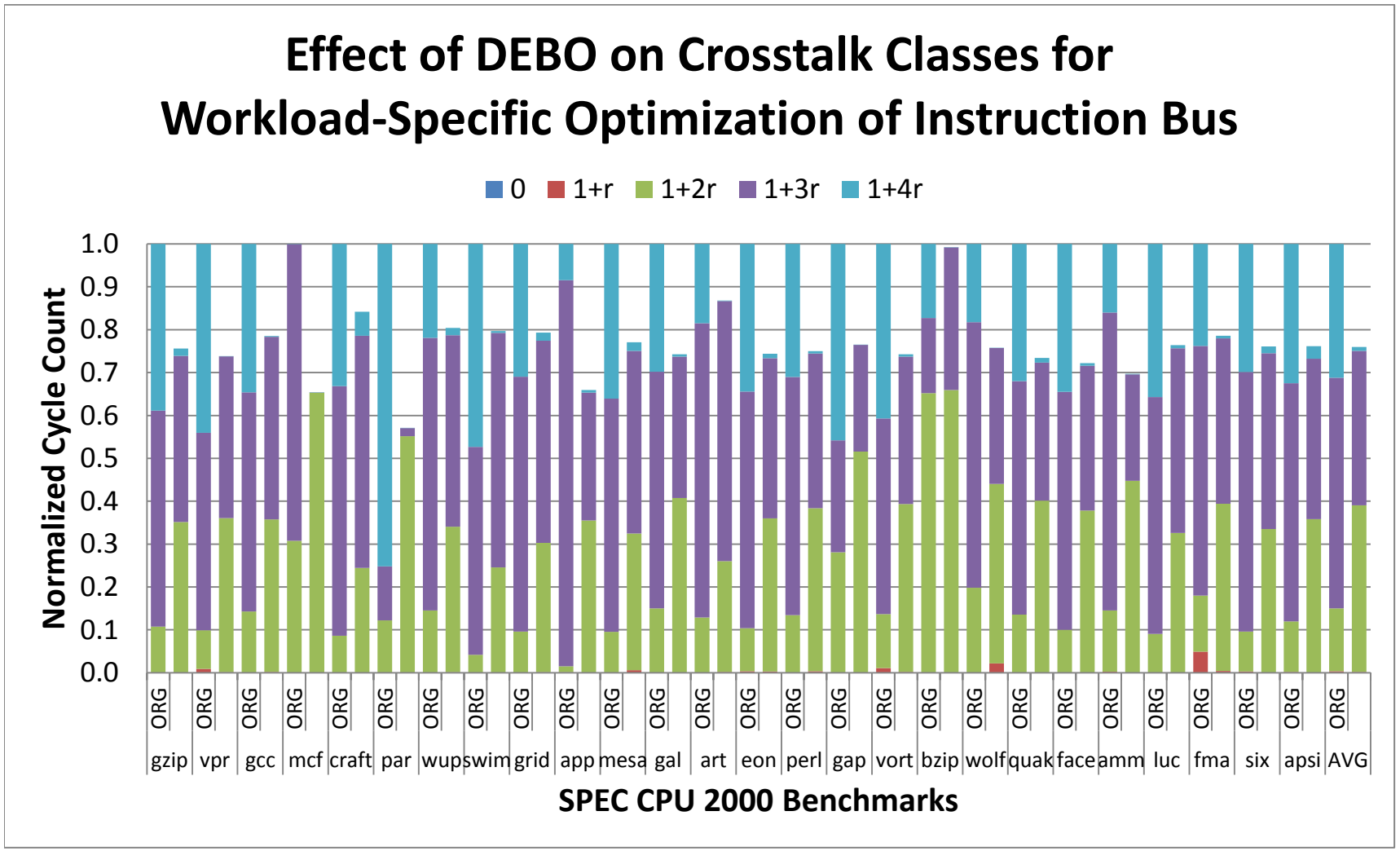


Figure 6.4: **Workload-Specific Performance Optimization of Instruction Traffic:** Percentage reduction in cycles with respect to original uncoded bus (ORG) using simultaneous bit ordering and dynamic encoding (DEBO) for secondary sample of SPEC CPU 2000 benchmark programs.

Chapter 7

Low-Power Adaptive Encoding for On-Chip Interconnects

7.1 Introduction

Efforts to reduce energy dissipation, particularly in long global address, instruction, and data buses, is becoming increasingly important in nanometer-scale technologies as interconnects continue to aggravate performance, power, and cost concerns. More than 50% of the total dynamic power dissipation in Intel processor was due to interconnects [7] and this is expected to rise to in modern architectures, such as multi-core, system-on-chip (SOC), and network-on-chip (NOC) [8].This increased energy/power dissipation also has an adverse effect on the temperature, reliability, performance, and cost of digital ICs. With technology scaling, the aspect ratio of global interconnects has been gradually increased to check growing resistance and delay of wires relative to those of transistors. However, this causes coupling capacitance between adjacent bus lines to become more pronounced. Solutions like introduction of low-k dielectrics between global interconnects layers can reduce coupling capacitance to some degree but are not sufficient. Further, introduction of low-k dielectrics aggravates the problem of wire self-heating [11].

Dynamic bus encoding is a widely-used approach exploiting the data-dependent nature of bus dynamic energy dissipation in which data is transmitted in an encoded form to save energy. Since real-world workloads cause bus traffic that exhibits significant spatial, temporal, and value locality, encoding schemes that exploit the characteristics of such correlated traffic can be very effective in improving bus energy efficiency. However, most existing low-power bus encoding schemes [65, 66, 88], which typically save energy by fully or partly inverting bus data opportunistically in selected cycles to reduce bus switching activity, are oblivious of these characteristics, and are effective only for random or worse-case (highly-changing) traffic. For realistic correlated data streams and, especially, wide (e.g., 64-bit or more) buses, these encoding schemes do not provide appreciable benefits since energy-saving encoding modes are not triggered frequently [86, 85].

Though value-aware dynamic encoding techniques [88, 85] have been proposed to exploit these traffic characteristics, dynamic encoding in general is limited by the number of encoding modes chosen at design-time. Hence, dynamic encoding schemes can degrade interconnect energy if traffic characteristics are vastly different from design-time assumptions. Due to temporal, spatial, and value locality only a small fraction of the possible values arise frequently during program execution [91, 170, 171, 172]. Hence, adaptive dynamic encoding scheme designed to choose encoding modes based on traffic characteristics can be far more effective than dynamic encoding schemes with static encoding modes. Previous adaptive dynamic encoding techniques were based on a small cache [91, 92] storing previously transmitted value or based on look-up table (LUT) [173] which load a predefined set of values before program execution. However, these techniques are limited to reducing self-transition and/or applicable to either data or instruction traffic.

7.1.1 Key Contributions and Results

In this chapter, we present an adaptive encoding scheme (AES) that maximizes energy savings by adapting the encoding mode based on traffic characteristics. The encoding modes

are stored in a small cache structure where each encoding mode indicates a set of bits to be inverted if that encoding mode is chosen. We present a novel replacement policy, based on partial-matches, where least recently used (LRU) encoding mode is replaced, when Hamming distance between data transmitted and encoding mode chosen to transmit the data exceed a certain threshold. The control information is also transmitted in a novel way where encoding mode is indicated using least recently used index rather than the index of the cache location where the encoding mode is store in order to reduce transition activity.

The efficacy of the AES approach depends on number of control lines (or encoding modes) and the threshold for the replacement policy. The number of encoding modes and threshold for the AES is determined using data collected from real-world programs, such as SPEC CPU2K benchmarks, executed using SimpleScalar/Alpha microarchitectural-level simulator. The AES scheme was tested on secondary traffic of each benchmark was able to reduce 30.90% of interconnect energy on data bus and 30.27% on instruction bus. In comparison, dynamic encoding schemes such as partitioned hybrid encoding (PHE) [85] was able to reduce only 29.55% and 24.22% for data and instruction buses, respectively, for program-specific optimization, where the encoding scheme designed using a benchmarks primary sample was applied the same benchmarks secondary sample.

Next, in Section 7.2, we present related work. Then, describe our approach to designing adaptive encoding scheme (AES) in Section 7.3 and discuss our results in Section 7.4. Finally, we conclude in Section 7.5.

7.2 Related Work

Adaptive or cache-based encoding schemes employs a cache structure to collect traffic characteristics based on history and determine efficient ways to transmit data. Though most of the works in data encoding address interconnect energy, earlier work addresses performance, area and throughput of buses, especially address buses, through the use of cache-based en-

coding schemes, such as dynamic base register caching (DBRC) and bus expander (BE) [67, 68]. These techniques used small compression caches at the sending end and register files at the receiving end of a processor-to-memory address bus and were first proposed to reduce off-chip bus widths (area) and pin count. These techniques were subsequently used in compression of on-chip instruction and data streams [69] to improve throughput. These compression techniques were also extended to address interconnect delay [70] and energy [71]. These cache-based encoding were also shown to be relevant particularly in the context of Network-on-Chip where interconnect is expected to dominate overall chip energy [8, 72, 73]. A perspective for optimizing performance, energy, and area/cost of these techniques were shown to yield significant improvements for address buses which were expected to be more pronounced for future technologies [74]. The DBRC was also extended to improve cost and performance, by exploiting value-characteristics, where partial matches in cache rather than a complete match were required for compression [75].

Cache-based adaptive encoding techniques, proposed for low-power interconnect design [91, 92], where reference values were updated based on traffic characteristics. However, to reduce encoder/decoder overhead due to the replacement policy, look-up table (LUT) based approaches were also proposed [173, 93, 89], where predefined reference values for a program were loaded before execution. However, cache-based adaptive encoding scheme reduces switching activity more effectively than LUT based approaches [91]. In addition, most of these adaptive techniques were proposed in the context of instruction, address or data bus.

7.3 Cache-Based Adaptive Encoding Scheme

Information transmitted in microprocessors show high degree of correlation across programs as well as across different phases of a program, due to the presence of temporal, spatial, and value localities. Temporal locality describes the likelihood that a recently referenced

item will be referenced again soon, while spatial locality describes the likelihood that a close neighbor of a recently referenced item will be referenced soon. Value locality, on the other hand, refers to the likelihood that a value produced by an instruction is the same as a value produced by another recently executed instruction. Hence, cache-based adaptive encoding schemes can be used to address major interconnect design concerns, by exploiting value locality in data traffic, and temporal and spatial locality in instruction traffic. The effectiveness of cache-based adaptive encoding scheme depends on the replacement policy, and size of the cache.

7.3.1 Signal and Control Line Transmission

The adaptive encoding scheme (AES) evaluate all encoding modes stored in the cache and identifies the mode that minimizes the interconnect-metric being addressed. While most cache-based encoding schemes use the cache index to indicate the control line scheme for the control line scheme we use the LRU index maintained both at the sender and receiver to signal the control line. Since, most recently used encoding mode is likely to be chosen, due to locality in programs, the LRU index with fewer ones could reduce the number of transitions that occur on the control line.

7.3.2 Cache Replacement Policy

The cache replacement policy influence how encoding modes dynamically adapt to changing traffic characteristics. Our cache replacement policy depends on how closely words are related to each other. The word to be transmitted is XOR'ed with each entry in the cache (equivalent to inverting a set of bits) and the most energy-efficient word is chosen for transmission. If the number of ones in the encoded word is less than a certain threshold then the cache access was considered a hit and the LRU indices's are updated based on the cache entry chosen, as shown in Figure 7.1. In case of a miss, the most energy efficient code is transmitted and the least recently used entry is replaced with original uncoded word that was transmitted and

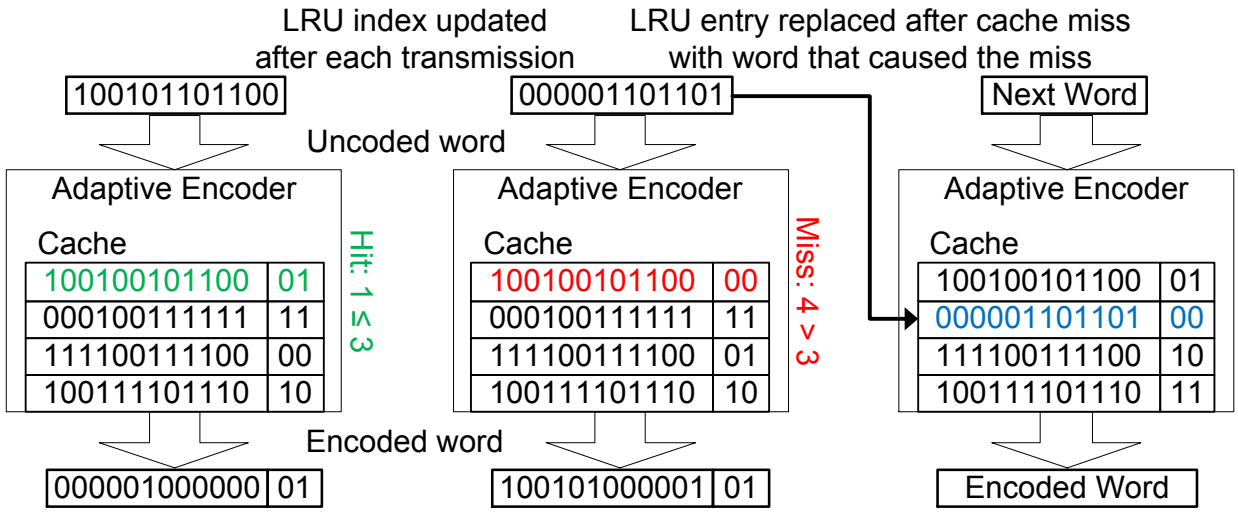


Figure 7.1: Adaptive encoding scheme showing how data is encoded using LRU index and how cache state is updated after a hit or a miss in each cycle.

the LRU index of the new entry set to zero. All other LRU indices's are updated accordingly.

7.4 Results and Discussion

In this section, we present results for our adaptive encoding scheme (AES) accounting for energy dissipated on the additional bus lines. The replacement threshold value and cache size were based on aggregate traffic characteristics of the primary sample of all 26 SPEC benchmarks and its efficacy tested by applying the scheme to the secondary sample of all 26 benchmarks. Figure 7.3 and Figure 7.2 shows the impact of replacement threshold and cache-size on energy savings of data and instruction traffic, respectively.

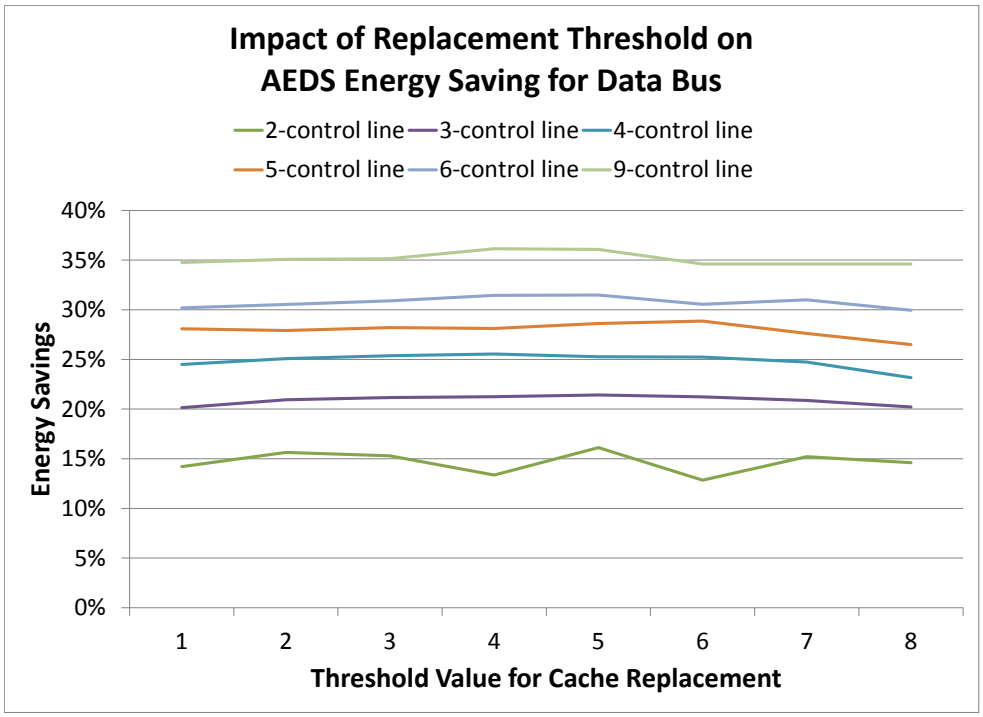
Energy savings across benchmarks for partitioned hybrid encoding (PHE), a value-aware dynamic encoding scheme was 25.43% and 28.64% for data and instruction buses, respectively. The results were obtained using program specific optimization, where PHE was optimized based on traffic characteristics of the primary sample of each individual benchmark and test on the secondary sample of that benchmark. We use program-specific optimization results of PHE because they are equivalent to LUT-based adaptive encoding where the encoding modes for PHE are values loaded into the look-up table before program execution.

In comparison, cache-based adaptive encoding technique with six control lines (which is the number of control lines used by PHE for instruction traffic) provides an energy savings of 30.90% and 30.27% for data and instruction traffic, respectively. The PHE data bus uses on average uses 16 control lines; however, we notice that our AES scheme provides better savings with fewer control lines. The energy savings across the programs are shown in Figure 7.4 and Figure 7.5 for data and instruction buses, respectively. In addition, the energy savings for AES instruction bus was in excess of 40% while the maximum energy savings realized by PHE was less than 30%.

Although AES schemes can be extended to exploit bit ordering we notice, for data traffic, that the energy savings of AES with *two* control lines is better than the dynamic encoding and bit ordering (DEBO) scheme presented in the previous chapter. The AES was able to reduce 30.00% of the self transitions compared to only 7.5% by the dynamic encoding and bit ordering scheme. This reduction is self transition helped reduce coupling energy which depends on the self-transition activity of two adjacent neighbors.

7.5 Conclusion

In this chapter, we presented a new adaptive encoding scheme (AES) in which encoding modes dynamically adapt to changing traffic characteristics to minimize interconnect energy. In contrast, previous adaptive encoding schemes were designed to work only for data or instruction traffic and did not account for coupling energy dissipation, while dynamic encoding technique cannot adapt to changing traffic characteristics. Hence, average bus energy savings across SPEC CPU2k benchmarks obtained using AES are in excess of 34% and 40% for both data and instruction buses, respectively. Compared to these results, energy savings using PHE was less than 30% over the same benchmarks for both data and instruction buses.



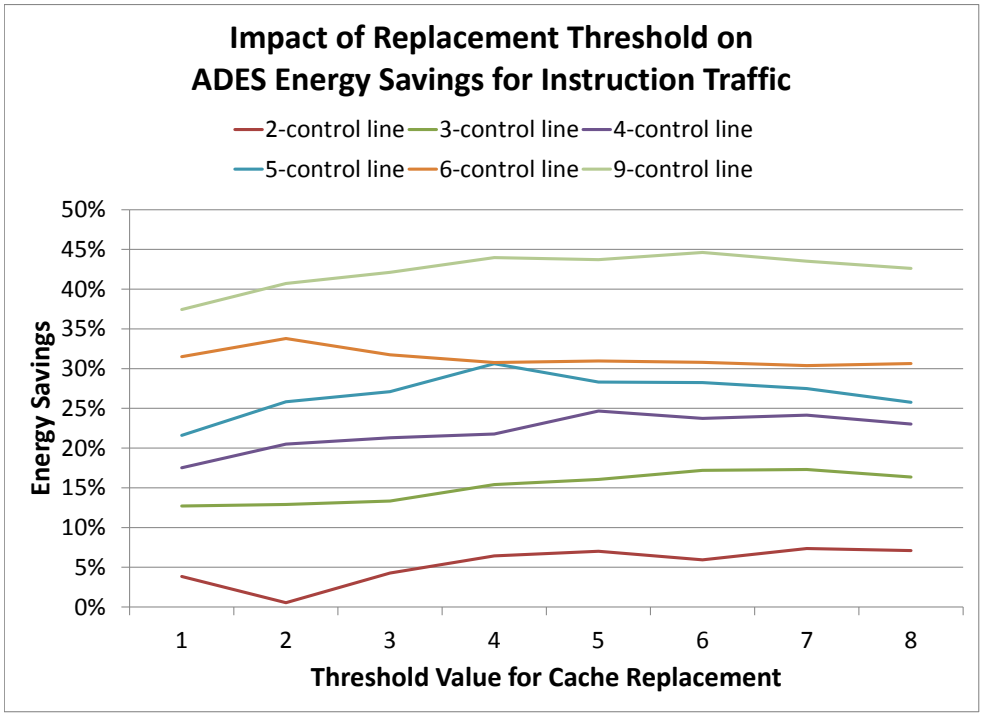
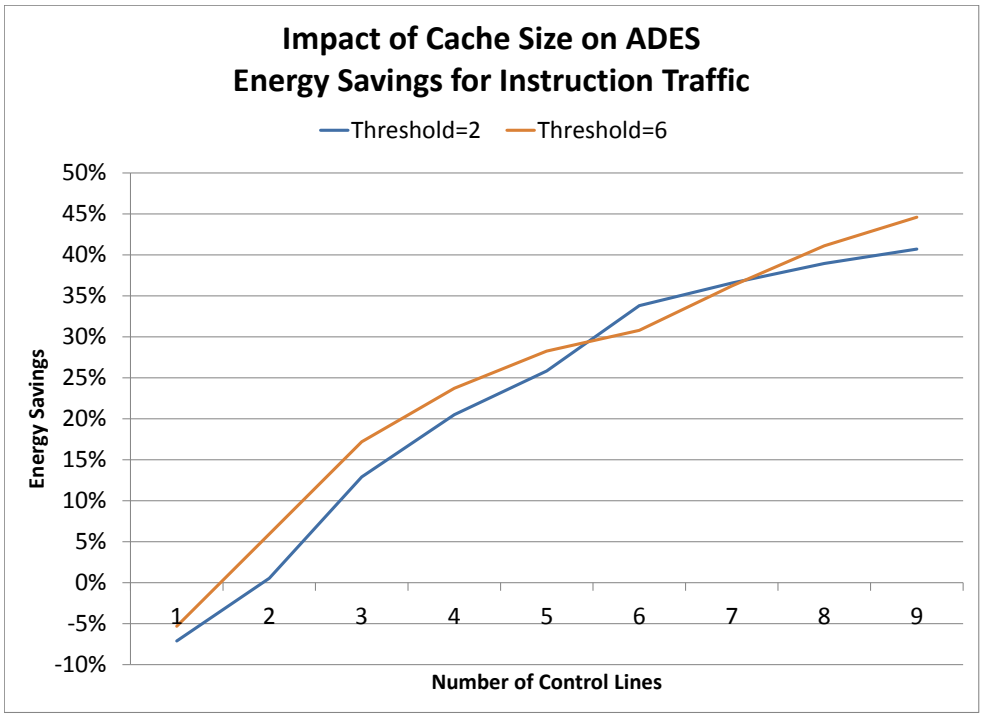


Figure 7.2: Effect of replacement threshold on the effectiveness of adaptive encoding scheme in reducing interconnect energy for data and instruction traffic, respectively.



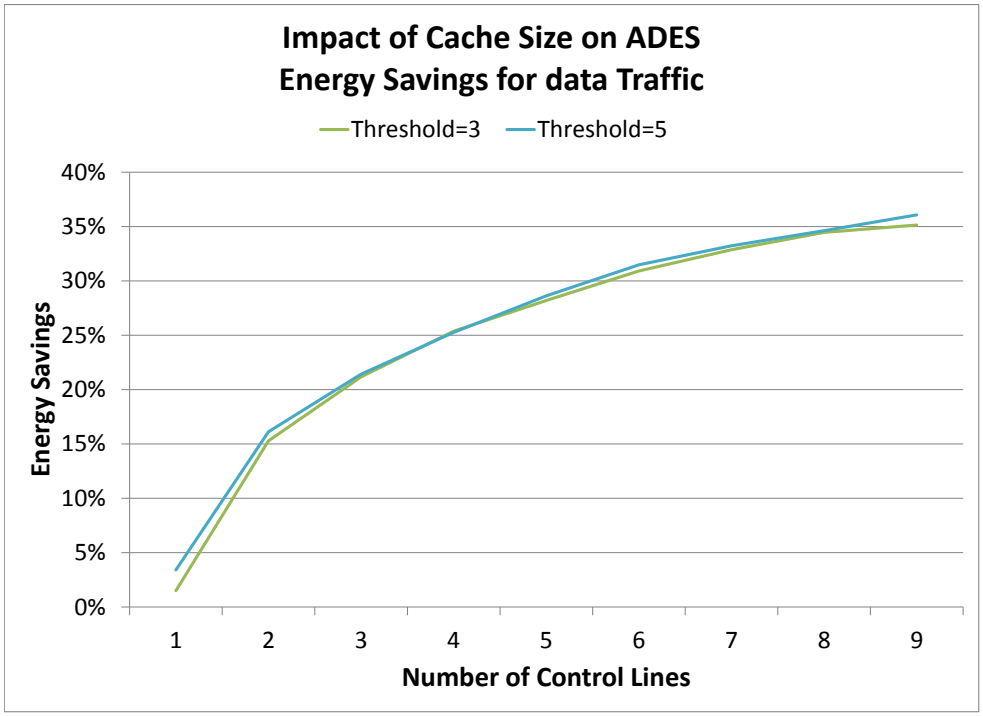


Figure 7.3: Effect of cache size on the effectiveness of adaptive encoding scheme in reducing interconnect energy for data and instruction traffic, respectively.

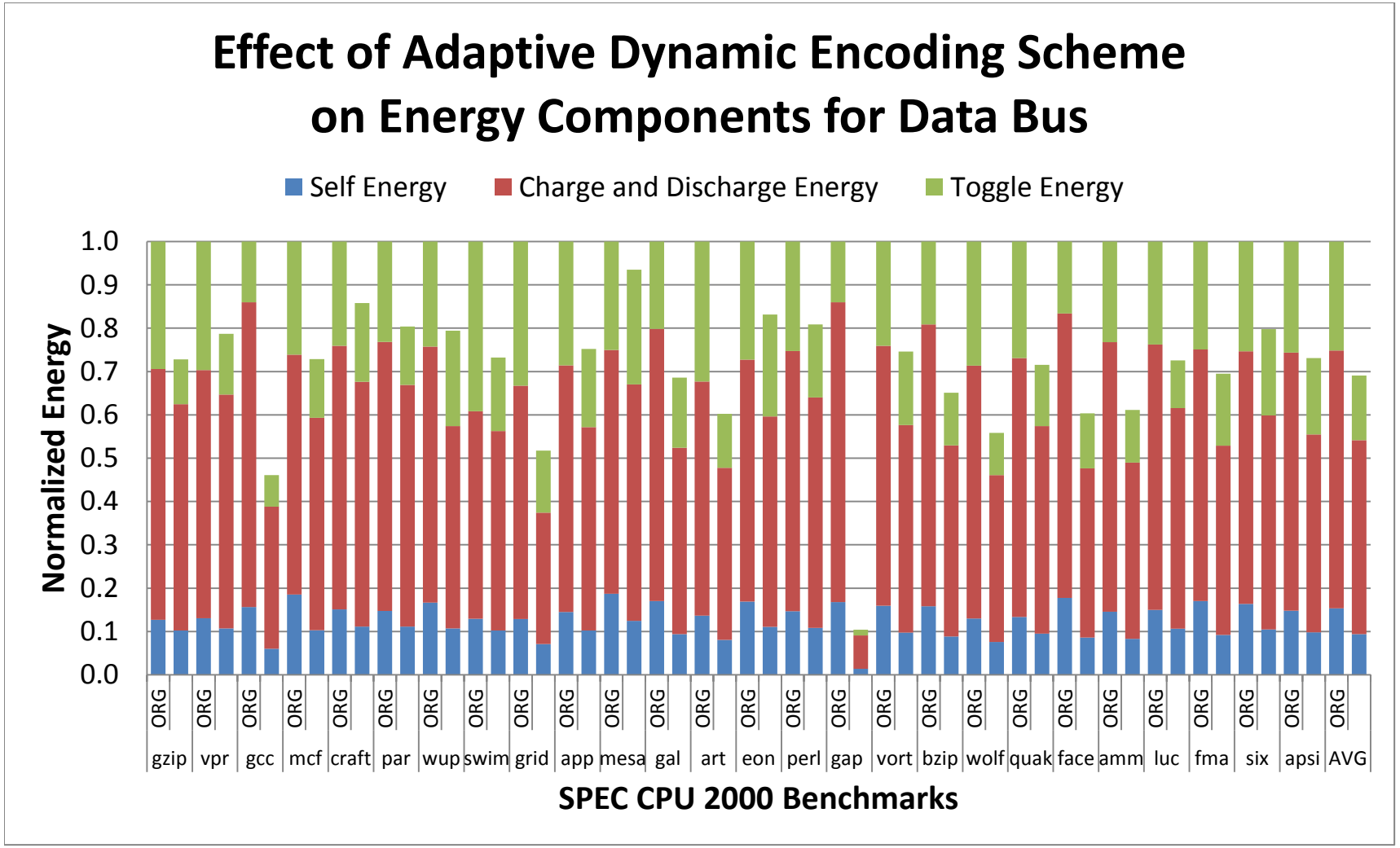


Figure 7.4: Data bus energy of SPEC CPU2K benchmark programs, using Adaptive encoding scheme with *six* control lines and replacement threshold value of *four* w.r.t original uncoded bus energy of that program sample.

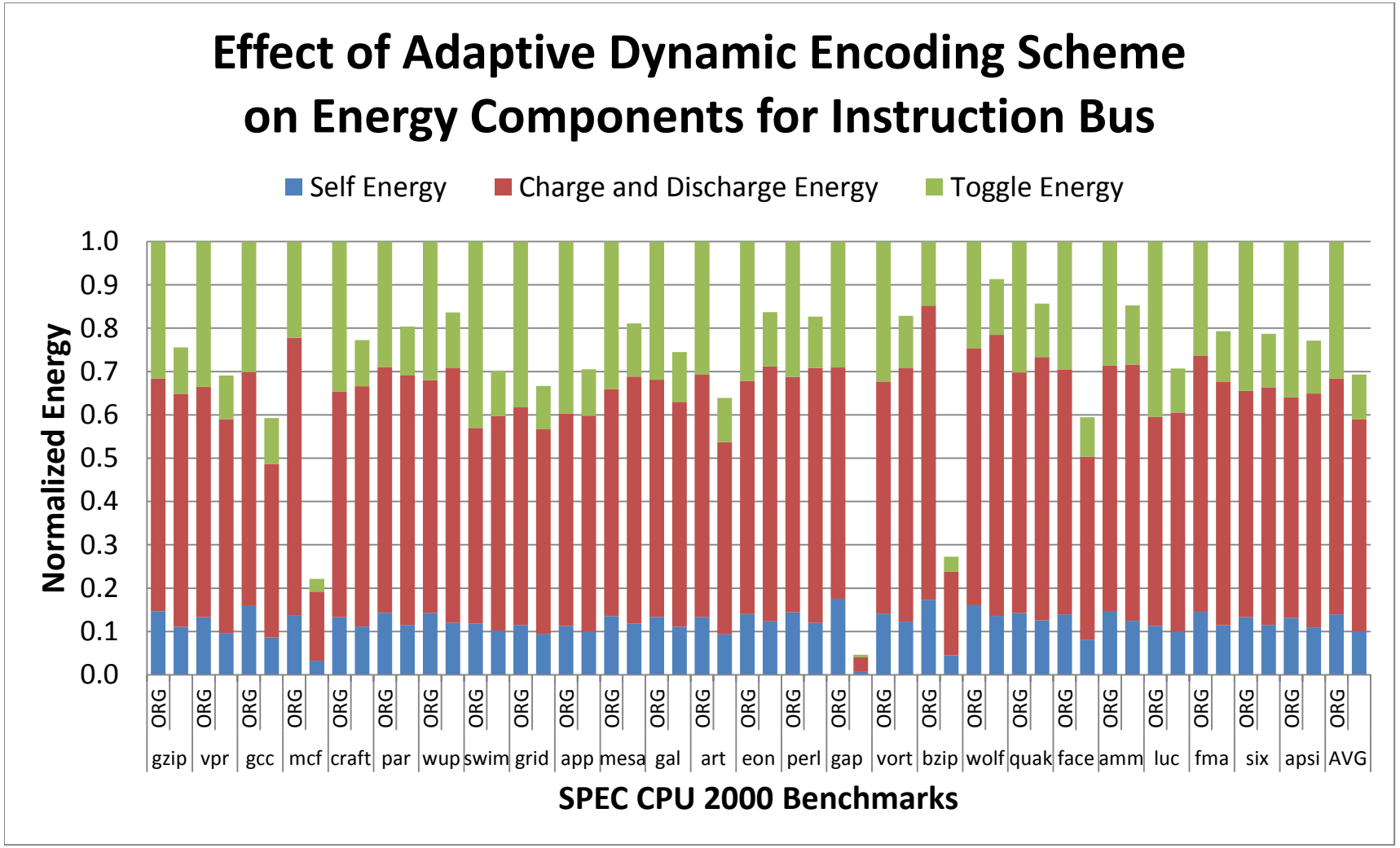


Figure 7.5: Instruction bus energy of SPEC CPU2K benchmark programs, using Adaptive encoding scheme with *six* control lines and replacement threshold value of *two* w.r.t original uncoded bus energy of that program sample.

Chapter 8

Conclusion

In this dissertation, we presented our research on value-aware multi-objective design for on-chip interconnects for current and future nanometer-scale technologies. We addressed three key interconnect issues (energy, performance, and temperature) using our ILP-based value-aware framework. Key contributions, results, and future work are described in this chapter.

8.1 Contributions and Key Results

In Chapter 4, we presented the efficacy of value-aware interconnect design by implementing two interconnect design techniques (dynamic bus encoding and wire spacing) based on traffic characteristics to maximize energy savings. The effectiveness of both techniques depends on additional area overhead available to accommodate additional signal wires or increased inter-wire spacing. The partitioned hybrid encoding scheme (PHE) designed using workload-specific optimization reduced interconnect energy by at most 22% and 24% for data and instruction buses, respectively. While, for value-aware wire spacing (VAWS), energy reduction was 39% and 25%, respectively, for area overhead comparable to PHE. The area overhead for energy-optimal PHE was 28% and 18% for data and instruction buses, respectively. For data bus we observed that the decrease in PHE energy was less than 0.02% for every addi-

tional control line beyond the first 10. Hence, VAWS performs better than PHE, however for lower area overhead the energy savings for both techniques were comparable. VAWS could increase Joule heating due to decreased metal to dielectric ratio which degrades wire delay and electromigration reliability. On the other hand, bus encoding techniques can be applied to current and future interconnect-technologies and can be used to address performance, energy, or reliability.

In Chapter 5, we presented an integer linear programming (ILP) framework to design a static encoding scheme that can address multiple interconnect objectives, simultaneously. The ILP framework was used to design a static encoding scheme where each bit was signaled using a static signaling scheme, selected from a group of signaling options, while simultaneously choosing a bit permutation based on bus traffic characteristics. The simultaneous bit ordering and signaling (SBOS) scheme optimized for energy was able to provide energy savings of 7.5% and 15% for data and instruction buses, respectively. We also presented a novel way to reduce the number of cycles with worst-case crosstalk to improve the performance of variable cycle transmission bus. The performance optimized SBOS designed for variable cycle bus transmission able to reduce the number of transmission cycles by 21.24% and 10.90% for data and instruction buses, respectively. Using the steady state thermal model we demonstrate that the peak temperature on the data bus using energy-optimal SBOS for *lucas* benchmark was within 1°K., even though energy dissipation was reduced by 24%. Using our proposed multi-objective approach for energy-temperature optimization, we were able to reduce peak wire temperature by more than 1°K for around 2% increase in optimal energy. Thus, showing the capability of our ILP-framework in addressing multiple interconnect objectives.

In Chapter 6, we developed a dynamic encoding scheme where an encoding mode is chosen, from a predefined set of encoding modes based on traffic characteristics, to minimize interconnect design metric. We classify the transition patterns using clustering and determine our encoding modes and a single bit permutation for the dynamic encoding scheme,

using the ILP-framework, to minimize our design objective. We reclassify transition patterns in the trace based on the encoding mode chosen for transmission and obtained a new dynamic encoding scheme using the ILP. When the energy reduction between two consecutive iterations was less than 1% this iterative procedure was stopped. To improve our overall solution time, we also proposed a heuristic approach that provides near optimal dynamic encoding scheme. The dynamic encoding and bit ordering scheme (DEBO) was able to reduce interconnect energy by 15.04% and 19.55% for data and instruction traffic, respectively. The increased flexibility in encoding increases energy savings by 1.3-2 times compared to SBOS, while having a positive impact on energy across all benchmark. The performance optimized DEBO was able to eliminate 17.18% and 24.00% of transmission cycles for variable cycle data and instruction buses. We see that increased encoding flexibility of dynamic encoding improved the interconnect design metrics. While number of cycles reduced more than doubled for performance optimized DEBO for instruction traffic, performance optimized DEBO, on an average, performed slightly worse than the static scheme. However, DEBO being a dynamic scheme was able to positively influence all benchmark, while SBOS degraded the performance of some benchmarks.

In Chapter 7, we increased the encoding freedom by adapting the set of dynamic encoding modes used to changing traffic characteristics. The adaptive dynamic encoding scheme (ADES) which uses a small fully associative cache was able to reduce interconnect energy 34% and 43% for data and instruction traffic, respectively. This is significantly greater than previous techniques, such as PHE and VAWS. Also, data bus using ADES performs better than DEBO, for the same number of control lines overhead. However, DEBO for instruction bus performs better than ADES for the same number of control line. However, most of the energy savings in DEBO was realized through bit reordering.

Our work represents a significant advancement over existing interconnect design approaches that consider worst-case or random traffic conditions. We also show the impact of encoding techniques on multiple interconnect objectives. In addition, our ILP-framework

provides an elegant approach to multi-objective interconnect design that can be easily integrated with existing design tools.

8.2 Future Research Directions

Some potential future research directions are outlined next.

- Combining the ILP framework for interconnect design with ILP-based approaches for optimizing gate sizing, and threshold voltage assignment and providing a methodology for chip design based on a single unified framework.

- Extend ILP framework to design encoding technique for other architecture, such as network-on-Chip, multi-core, or system-on-chip, where traffic characteristics are influenced not only by the data but by also network protocols and multiple programs executing.

- Model data dependent nature of newer interconnect technologies, such as carbon nanotubes, optical interconnect, 3D ICs to be incorporated into our ILP framework to design an encoding.

- Creating configurable interconnect intellectual property (IIP) blocks suitably optimized for power, temperature, and crosstalk for user specified workloads, which can then be easily synthesized by CAD tools.

Bibliography

Bibliography

[1] Semiconductor Industry Association, "International Technology Roadmap for Semiconductors (ITRS), 2009 Edition." URL: `http://public.itrs.net`, 2009.

[2] S. Borkar, "Thousand core chips: A technology perspective," in *Proceedings of the Annual ACM/IEEE Design Automation Conference*, (New York, NY, USA), pp. 746–749, ACM, 2007.

[3] L. Lev and S. Ping Chao, "Down to the wire: Requirements for nanometer design implementation," *Cadence White Papers, URL: http://www.eetimes.com/news/design/showArticle.jhtml?articleID=16505500*, 2002.

[4] S. Rusu, "Circuit technologies for multi-core design." URL: `http://www.ewh.ieee.org/r6/scv/ssc/April06.pdf`, 2006.

[5] J. A. Davis and J. D. Meindl, *Interconnect Technology and Design for Gigascale Integration*. Norwell, MA, USA: Kluwer Academic Publishers, 2003.

[6] M. Bamal, Y. Travaly, W. Zhang, M. Stucchi, and K. Maex, "Impact of interconnect resistance increase on system performance of low power and high performance designs," in *Proceedings of the International Workshop on System Level Interconnect Prediction*, (New York, NY, USA), pp. 85–90, ACM, 2006.

[7] N. Magen, A. Kolodny, U. Weiser, and N. Shamir, "Interconnect-power dissipation in a microprocessor," in *Proceedings of the International Workshop on System Level Interconnect Prediction*, pp. 7–13, ACM Press, 2004.

[8] A. Flores, J. L. Aragón, and M. E. Acacio, "An energy consumption characterization of on-chip interconnection networks for tiled cmp architectures," *Springer Journal of Supercomputing*, vol. 45, no. 3, pp. 341–364, 2008.

[9] S. Im and K. Banerjee, "Full chip thermal analysis of planar (2-D) and vertically integrated (3-D) high performance ICs," *Technical Digest of the International Electron Devices Meeting*, pp. 727–730, 2000.

[10] S. Im, N. Srivastava, K. Banerjee, and K. E. Goodson, "Scaling analysis of multilevel interconnect temperatures for high-performance ICs," *IEEE Transactions on Electron Devices*, vol. 52, pp. 2710 – 2719, dec. 2005.

[11] K. Banerjee, "Trends for ULSI interconnections and their implications for thermal, reliability and performance issues (invited paper)," in *Proceedings of the Seventh International Dielectrics and Conductors for ULSI Multilevel Interconnection Conference*, pp. 38–50, Mar 2001.

[12] A. H. Ajami, K. Banerjee, and M. Pedram, "Modeling and analysis of nonuniform substrate temperature effects on global ULSI interconnects," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, pp. 849–861, June 2005.

[13] R. P. Dick, "Reliability, thermal, and power modeling and optimization," in *Proceedings of the International Conference on Computer-Aided Design*, pp. 181 –184, nov. 2010.

[14] Z. Lu, W. Huang, M. R. Stan, K. Skadron, and J. Lach, "Interconnect lifetime prediction for reliability-aware systems," *IEEE Transactions on VLSI Systems*, vol. 15, pp. 159–172, Feb. 2007.

[15] W. Huang, M. R. Stan, K. Skadron, K. Sankaranarayanan, S. Ghosh, S. Ghosh, and S. Velusam, "Compact thermal modeling for temperature-aware design," in *Proceedings of the Annual ACM/IEEE Design Automation Conference*, (New York, NY, USA), pp. 878–883, ACM, 2004.

[16] J. Gambino, F. Chen, and J. He, "Copper interconnect technology for the 32 nm node and beyond," in *Proceedings of the IEEE Custom Integrated Circuits Conference*, pp. 141 –148, sept. 2009.

[17] P. Zarkesh-Ha, P. Bendix, W. Loh, J. Lee, and J. Meindl, "The impact of Cu/low-k on chip performance," *Proceedings of IEEE International ASIC/SOC Conference*, pp. 257–261, 1999.

[18] A. Naeemi and J. D. Meindl, "Design and performance modeling for single-walled carbon nanotubes as local, semiglobal, and global interconnects in gigascale integrated systems," *IEEE Transactions on Electron Devices*, vol. 54, pp. 26–37, Jan. 2007.

[19] J. W. Joyner, R. Venkatesan, P. Zarkesh-Ha, J. A. Davis, and J. D. Meindl, "Impact of three-dimensional architectures on interconnects in gigascale integration," *IEEE Transactions on VLSI Systems*, vol. 9, no. 6, pp. 922–928, 2001.

[20] J. D. Meindl, "Interconnect opportunities for gigascale integration," *IEEE Micro*, vol. 23, no. 3, pp. 28–35, 2003.

[21] H. Zhou, J. Flanagan, and T. M. Conte, "Detecting global stride locality in value streams," in *Proceedings of the Annual Symposium on Computer Architecture*, (New York, NY, USA), pp. 324–335, ACM, 2003.

[22] J. Yang and R. Gupta, "Frequent value locality and its applications," *ACM Transactions on Embedded Computing Systems*, vol. 1, no. 1, pp. 79–105, 2002.

[23] S. Balakrishnan and G. S. Sohi, "Exploiting value locality in physical register files," in *Proceedings of the Annual ACM/IEEE International Symposium on Microarchitecture*, (Washington, DC, USA), pp. 265–276, IEEE Computer Society, 2003.

[24] A. H. Farrahi, C. Chen, A. Srivastava, G. Tellez, and M. Sarrafzadeh, "Activity-driven clock design," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 20, pp. 705–714, June 2001.

[25] Z. Hu, A. Buyuktosunoglu, V. Srinivasan, V. Zyuban, H. Jacobson, and P. Bose, "Microarchitectural techniques for power gating of execution units," in *Proceedings of the International Symposium on Low Power Electronics and Design*, (New York, NY, USA), pp. 32–37, ACM, 2004.

[26] O. Golubeva, M. Loghi, E. Macii, and M. Poncino, "Locality-driven architectural cache sub-banking for leakage energy reduction," in *Proceedings of the International Symposium on Low Power Electronics and Design*, (New York, NY, USA), pp. 274–279, ACM, 2007.

[27] M. H. Lipasti, C. B. Wilkerson, and J. P. Shen, "Value locality and load value prediction," in *Proceedings of Architectural Support for Programming Languages and Operating Systems*, (New York, NY, USA), pp. 138–147, ACM, 1996.

[28] V. Raghunathan, M. B. Srivastava, and R. K. Gupta, "A survey of techniques for energy efficient on-chip communication," in *Proceedings of the Annual ACM/IEEE Design Automation Conference*, (New York, NY, USA), pp. 900–905, ACM, 2003.

[29] L. Benini and G. de Micheli, "System-level power optimization: techniques and tools," *ACM Transactions on Design Automation of Electronic Systems*, vol. 5, no. 2, pp. 115–192, 2000.

[30] C.-Y. Wu and M.-C. Shiau, "Delay models and speed improvement techniques for RC tree interconnections among small-geometry CMOS inverters," *IEEE Journal of Solid-State Circuits*, vol. 25, pp. 1247–1256, Oct 1990.

[31] M. Nekili and Y. Savaria, "Optimal methods of driving interconnections in VLSI circuits," in *Proceedings of IEEE International Symposium on Circuits and Systems*, vol. 1, pp. 21–24, May 1992.

[32] S. Dhar and M. A. Franklin, "Optimum buffer circuits for driving long uniform lines," *IEEE Journal of Solid-State Circuits*, vol. 26, pp. 32–40, Jan 1991.

[33] L. P. P. P. van Ginneken, "Buffer placement in distributed RC-tree networks for minimal Elmore delay," in *Proceedings of IEEE International Symposium on Circuits and Systems*, pp. 865–868, 1990.

[34] J. Lillis, C.-K. Cheng, and T.-T. Y. L. Lin, "Optimal wire sizing and buffer insertion for low power and a generalized delay model," *IEEE Journal of Solid-State Circuits*, vol. 31, no. 3, pp. 437–447, Mar 1996.

[35] C. Alpert and A. Devgan, "Wire segmenting for improved buffer insertion," in *Proceedings of the Annual ACM/IEEE Design Automation Conference*, pp. 588–593, Jun 1997.

[36] V. Adler and E. G. Friedman, "Uniform repeater insertion in RC trees," *IEEE Transactions on Circuits and Systems 1: Fundamental Theory and Applications*, vol. 47, no. 10, pp. 1515–1523, 2000.

[37] S. Srinivasaraghavan and W. Burleson, "Interconnect effort - a unification of repeater insertion and logical effort," in *Proceedings of the IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, (Washington, DC, USA), pp. 55–60, IEEE Computer Society, 2003.

[38] D. Li, A. Pua, P. Srivastava, and U. Ko, "A repeater optimization methodology for deep sub-micron, high-performance processors," in *Proceedings of the IEEE International Conference on Computer Design*, (Washington, DC, USA), pp. 726–731, IEEE Computer Society, 1997.

[39] Y. I. Ismail and E. G. Friedman, "Optimum repeater insertion based on a CMOS delay model for on-chip RLC interconnect," in *Proceedings of IEEE International ASIC/SOC Conference*, pp. 369–373, Sep 1998.

[40] A. B. Kahng, S. Muddu, E. Sarto, and R. Sharma, "Interconnect tuning strategies for high-performance ICs," in *Proceedings of the Conference on Design, Automation and Test in Europe*, (Washington, DC, USA), pp. 471–478, IEEE Computer Society, 1998.

[41] C. J. Akl and M. A. Bayoumi, "Reducing delay uncertainty of on-chip interconnects by combining inverting and non-inverting repeaters insertion," in *Proceedings of the International Symposium on Quality Electronic Design*, (Washington, DC, USA), pp. 219–224, IEEE Computer Society, 2007.

[42] R. Weerasekera, D. Pamunuwa, L.-R. Zheng, and H. Tenhunen, "Minimal-power, delay-balanced smart repeaters for interconnects in the nanometer regime," in *Proceedings of the International Workshop on System Level Interconnect Prediction*, (New York, NY, USA), pp. 113–120, ACM, 2006.

[43] M. Khellah, M. Ghoneima, J. Tschanz, Y. Ye, N. Kurd, J. Barkatullah, S. Nimmagadda, and Y. Ismail, "A skewed repeater bus architecture for on-chip energy reduction in microprocessors," in *Proceedings of the IEEE International Conference on Computer Design*, (Washington, DC, USA), pp. 253–257, IEEE Computer Society, 2005.

[44] M. Ghoneima and Y. I. Ismail, "Optimum positioning of interleaved repeaters in bidirectional buses," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, no. 3, pp. 461–469, 2005.

[45] W. T. Cheung and N. Wong, "Power optimization in a repeater-inserted interconnect via geometric programming," in *Proceedings of the International Symposium on Low Power Electronics and Design*, (New York, NY, USA), pp. 226–231, ACM Press, 2006.

[46] Y. Peng and X. Liu, "Low-power repeater insertion with both delay and slew rate constraints," in *Proceedings of the Annual ACM/IEEE Design Automation Conference*, (New York, NY, USA), pp. 302–307, ACM Press, 2006.

[47] J. Xiong and L. He, "Fast buffer insertion considering process variations," in *Proceedings of the International Symposium on Physical Design*, (New York, NY, USA), pp. 128–135, ACM, 2006.

[48] J. Cong, K.-S. Leung, and D. Zhou, "Performance-driven interconnect design based on distributed RC delay model," in *Proceedings of the Annual ACM/IEEE Design Automation Conference*, (New York, NY, USA), pp. 606–611, ACM, 1993.

[49] J. Cong and K.-S. Leung, "Optimal wire sizing under the distributed Elmore delay model," in *Proceedings of the International Conference on Computer-Aided Design*, (Los Alamitos, CA, USA), pp. 634–639, IEEE Computer Society Press, 1993.

[50] J. P. Fishburn and C. A. Schevon, "Shaping a distributed-RC line to minimize Elmore delay," *IEEE Transactions on Circuits and Systems 1: Fundamental Theory and Applications*, vol. 42, no. 12, pp. 1020–1022, 1995.

[51] C.-P. Chen, Y.-P. Chen, and M. D. F. Wong, "Optimal wire-sizing formula under the Elmore delay model," in *Proceedings of the Annual ACM/IEEE Design Automation Conference*, (New York, NY, USA), pp. 487–490, ACM Press, 1996.

[52] J. J. Cong and K.-S. Leung, "Optimal wiresizing under Elmore delay model," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 14, pp. 321–336, Mar 1995.

[53] S. S. Sapatnekar, "RC interconnect optimization under the Elmore delay model," in *Proceedings of the Annual ACM/IEEE Design Automation Conference*, (New York, NY, USA), pp. 387–391, ACM, 1994.

[54] C.-P. Chen and M. D. F. Wong, "A fast algorithm for optimal wire-sizing under Elmore delay model," in *Proceedings of IEEE International Symposium on Circuits and Systems*, vol. 4, pp. 412–415, 1996.

[55] C.-P. Chen, H. Zhou, and M. D. F. Wong, "Optimal non-uniform wire-sizing under the Elmore delay model," in *Proceedings of the International Conference on Computer-Aided Design*, (Washington, DC, USA), pp. 38–43, IEEE Computer Society, 1996.

[56] J. Cong, L. He, C.-K. Koh, and Z. Pan, "Global interconnect sizing and spacing with consideration of coupling capacitance," in *Proceedings of the International Conference on Computer-Aided Design*, (Washington, DC, USA), pp. 628–633, IEEE Computer Society, 1997.

[57] N. Menezes, R. Baldick, and L. T. Pileggi, "A sequential quadratic programming approach to concurrent gate and wire sizing," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 16, pp. 867–881, Aug 1997.

[58] C. Chu and M. D. F. Wong, "Closed form solutions to simultaneous buffer insertion/sizing and wire sizing," *ACM Transactions on Design Automation of Electronic Systems*, vol. 6, no. 3, pp. 343–371, 2001.

[59] C.-P. Chen, C. C. N. Chu, and M. D. F. Wong, "Fast and exact simultaneous gate and wire sizing by Lagrangian relaxation," in *Proceedings of the International Conference on Computer-Aided Design*, (New York, NY, USA), pp. 617–624, ACM, 1998.

[60] L. He, A. Kahng, K. H. Tam, and J. Xiong, "Simultaneous buffer insertion and wire sizing considering systematic CMP variation and random L_{eff} variation," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 26, no. 5, pp. 845–857, 2007.

[61] Y. Gao and M. D. F. Wong, "Optimal wire shape with consideration of coupling capacitance under Elmore delay model," in *Proceedings of Asia and South Pacific Design Automation Conference*, vol. 1, pp. 217–220, Jan 1999.

[62] M. A. El-Moursy and E. G. Friedman, "Optimum wire shaping of an RLC interconnect," in *Proceedings of the IEEE International Midwest Symposium on Circuits and Systems*, vol. 3, pp. 1459–1464, 2003.

[63] I. H.-R. Jiang, Y.-W. Chang, and J.-Y. Jou, "Crosstalk-driven interconnect optimization by simultaneous gate and wire sizing," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 19, pp. 999–1010, Sep 2000.

[64] J. Cong and Z. Pan, "Wire width planning for interconnect performance optimization," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 21, pp. 319–329, Mar 2002.

[65] M. R. Stan and W. P. Burleson, "Bus-invert coding for low-power I/O," *IEEE Transactions on VLSI Systems*, vol. 3, no. 1, pp. 49–58, 1995.

[66] Y. Zhang, J. Lach, K. Skadron, and M. R. Stan, "Odd/even bus invert with two-phase transfer for buses with coupling," in *Proceedings of the International Symposium on Low Power Electronics and Design*, (New York, NY, USA), pp. 80–83, ACM, 2002.

[67] A. Park and M. Farrens, "Address compression through base register caching," in *Proceedings of the Annual ACM/IEEE International Symposium on Microarchitecture*, (Los Alamitos, CA, USA), pp. 193–199, IEEE Computer Society Press, 1990.

[68] M. Farrens and A. Park, "Dynamic base register caching: a technique for reducing address bus width," in *Proceedings of the Annual Symposium on Computer Architecture*, (New York, NY, USA), pp. 128–137, ACM, 1991.

[69] D. Citron and L. Rudolph, "Creating a wider bus using caching techniques," in *Proceedings of the International Symposium on High Performance Computer Architecture*, (Washington, DC, USA), pp. 90–99, IEEE Computer Society, 1995.

[70] D. Citron, "Exploiting low entropy to reduce wire delay," *IEEE Computer Architecture Letters*, vol. 3, pp. 1–4, Jan. 2004.

[71] J. Liu, K. Sundaresan, and N. R. Mahapatra, "Energy-efficient compressed address transmission," in *Proceedings of the International Conference on VLSI Design*, (Washington, DC, USA), pp. 592–597, IEEE Computer Society, 2005.

[72] Y. Jin, K. H. Yum, and E. J. Kim, "Adaptive data compression for high-performance low-power on-chip networks," in *Proceedings of the Annual ACM/IEEE International Symposium on Microarchitecture*, pp. 354 –363, nov. 2008.

[73] R. Das, A. K. Mishra, C. Nicopoulos, D. Park, V. Narayanan, R. Iyer, M. S. Yousif, and C. R. Das, "Performance and power optimization through data compression in network-on-chip architectures," in *Proceedings of the International Symposium on High Performance Computer Architecture*, pp. 215 –225, feb. 2008.

[74] J. Liu, K. Sundaresan, and N. R. Mahapatra, "Dynamic address compression schemes: A performance, energy, and cost study," in *Proceedings of the IEEE International Conference on Computer Design*, (Washington, DC, USA), pp. 458–463, IEEE Computer Society, 2004.

[75] J. Liu, K. Sundaresan, and N. R. Mahapatra, "Fast performance-optimized partial match compression for low-latency on-chip address buses," in *Proceedings of the IEEE International Conference on Computer Design*, pp. 17–24, oct. 2006.

[76] W. Fornaciari, M. Polentarutti, D. Sciuto, and C. Silvano, "Power optimization of system-level address buses based on software profiling," in *Proceedings of the IEEE/ACM International Conference on Hardware/Software Codesign and System Synthesis*, (New York, NY, USA), pp. 29–33, ACM, 2000.

[77] L. Benini, G. de Micheli, E. Macii, D. Sciuto, and C. Silvano, "Asymptotic zero-transition activity encoding for address busses in low-power microprocessor-based systems," in *Proceedings of the Great Lakes Symposium on VLSI*, (Washington, DC, USA), pp. 77–82, IEEE Computer Society, 1997.

[78] E. Musoll, T. Lang, and J. Cortadella, "Working-zone encoding for reducing the energy in microprocessor address buses," *IEEE Transactions on VLSI Systems*, vol. 6, no. 4, pp. 568–572, 1998.

[79] Y. Aghaghiri, F. Fallah, and M. Pedram, "Irredundant address bus encoding for low power," in *Proceedings of the International Symposium on Low Power Electronics and Design*, (New York, NY, USA), pp. 182–187, ACM, 2001.

[80] Y. Aghaghiri, M. Pedram, and F. Fallah, "Transition reduction in memory buses using sector-based encoding techniques," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 23, no. 8, pp. 1164–1174, 2004.

[81] M. R. Stan and W. P. Burleson, "Two dimensional codes for low power," in *Proceedings of the International Symposium on Low Power Electronics and Design*, (Piscataway, NJ, USA), pp. 335–340, IEEE Press, 1996.

[82] Y. Shin, K. Choi, and Y.-H. Chang, "Narrow bus encoding for low-power DSP systems," *IEEE Transactions on VLSI Systems*, vol. 9, no. 5, pp. 656–660, 2001.

[83] K. W. Kim, K. H. Baek, N. Shanbhag, C. L. Liu, and S. M. Kang, "Coupling-driven signal encoding scheme for low-power interface design," in *Proceedings of the International Conference on Computer-Aided Design*, (Piscataway, NJ, USA), pp. 318–321, IEEE Press, 2000.

[84] M. Ghoneima and Y. I. Ismail, "Low power coupling-based encoding for on-chip buses," in *Proceedings of IEEE International Symposium on Circuits and Systems*, pp. 325–328, 2004.

[85] S. Jayaprakash and N. R. Mahapatra, "Partitioned hybrid encoding to minimize on-chip energy dissipation of wide microprocessor buses," in *Proceedings of the International Conference on VLSI Design*, (Washington, DC, USA), pp. 127–134, IEEE Computer Society, 2007.

[86] K. Sundaresan and N. R. Mahapatra, "Accurate energy dissipation and thermal modeling for nanometer-scale buses," in *Proceedings of the International Symposium on High Performance Computer Architecture*, (Washington, DC, USA), pp. 51–60, IEEE Computer Society, 2005.

[87] A. Phansalkar, A. Joshi, and L. K. John, "Analysis of redundancy and application balance in the SPEC CPU2006 benchmark suite," in *Proceedings of the Annual Symposium on Computer Architecture*, (New York, NY, USA), pp. 412–423, ACM, 2007.

[88] Y. Shin, S.-I. Chae, and K. Choi, "Partial bus-invert coding for power optimization of application-specific systems," *IEEE Transactions on VLSI Systems*, vol. 9, no. 2, pp. 377–383, 2001.

[89] R. Siegmund, C. Kretzschmar, and D. Muller, "Adaptive partial bus invert encoding for power efficient data transfer over wide system buses," in *SBCCI: Proceedings of the Symposium on Integrated Circuits and Systems Design*, (Washington, DC, USA), pp. 371–376, IEEE Computer Society, 2000.

[90] P. Petrov and A. Orailoglu, "Transforming binary code for low-power embedded processors," *IEEE Micro*, vol. 24, pp. 21–33, May-June 2004.

[91] J. Yang, R. Gupta, and C. Zhang, "Frequent value encoding for low power data buses," *ACM Transactions on Design Automation of Electronic Systems*, vol. 9, no. 3, pp. 354–384, 2004.

[92] C. Liu, A. Sivasubramaniam, and M. Kandemir, "Optimizing bus energy consumption of on-chip multiprocessors using frequent values," *Elsevier Journal of Systems Architecture*, vol. 52, no. 2, pp. 129–142, 2006.

[93] T. Lv, J. Henkel, H. Lekatsas, and W. Wolf, "A dictionary-based en/decoding scheme for low-power data buses," *IEEE Transactions on VLSI Systems*, vol. 11, no. 5, pp. 943–951, 2003.

[94] Y. Shin and T. Sakurai, "Coupling-driven bus design for low-power application-specific systems," in *Proceedings of the Annual ACM/IEEE Design Automation Conference*, (New York, NY, USA), pp. 750–753, ACM, 2001.

[95] K. Sundaresan and N. R. Mahapatra, "Value-based bit ordering for energy optimization of on-chip global signal buses," in *Proceedings of the Conference on Design, Automation and Test in Europe*, (3001 Leuven, Belgium, Belgium), pp. 624–625, European Design and Automation Association, 2006.

[96] K. Sundaresan and N. R. Mahapatra, "Interconnect signaling and layout optimization to manage thermal effects due to self heating in on-chip signal buses," in *Proceedings of the International Symposium on Quality Electronic Design*, (Washington, DC, USA), pp. 118–122, IEEE Computer Society, 2008.

[97] S. Jayaprakash and N. R. Mahapatra, "Energy-optimal signaling and ordering of bits for area-constrained interconnects," in *Proceedings of IEEE International System on Chip (SOC) Conference*, pp. 9–12, Sept. 2008.

[98] B. Victor and K. Keutzer, "Bus encoding to prevent crosstalk delay," in *Proceedings of the International Conference on Computer-Aided Design*, (Piscataway, NJ, USA), pp. 57–63, IEEE Press, 2001.

[99] A. Bogliolo, "Encodings for high-performance for energy-efficient signaling," in *Proceedings of the International Symposium on Low Power Electronics and Design*, (New York, NY, USA), pp. 170–175, ACM, 2001.

[100] R. Hossain, F. Viglione, and M. Cavalli, "Designing fast on-chip interconnects for deep submicrometer technologies," *IEEE Transactions on VLSI Systems*, vol. 11, no. 2, pp. 276–280, 2003.

[101] S. R. Sridhara and N. R. Shanbhag, "Coding for system-on-chip networks: a unified framework," in *Proceedings of the Annual ACM/IEEE Design Automation Conference*, (New York, NY, USA), pp. 103–106, ACM, 2004.

[102] M. Lampropoulos, B. M. Al-Hashimi, and P. Rosinger, "Minimization of crosstalk noise, delay and power using a modified bus invert technique," in *Proceedings of the Conference on Design, Automation and Test in Europe*, (Washington, DC, USA), pp. 1372–1373, IEEE Computer Society, 2004.

[103] S. Gupta and S. Katkoori, "Intrabus crosstalk estimation using word-level statistics," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, pp. 469–478, March 2005.

[104] R. Ayoub and A. Orailoglu, "A unified transformational approach for reductions in fault vulnerability, power, and crosstalk noise & delay on processor buses," in *Proceedings of Asia and South Pacific Design Automation Conference*, vol. 2, (New York, NY, USA), pp. 729–734, ACM, Jan. 2005.

[105] K. Najeeb, V. Gupta, and V. Kamakoti, "Delay and peak power minimization for on-chip buses using temporal redundancy," in *Proceedings of the Great Lakes Symposium on VLSI*, (New York, NY, USA), pp. 119–122, ACM, 2006.

[106] W.-W. Hsieh, P.-Y. Chen, and T. Hwang, "A bus architecture for crosstalk elimination in high performance processor design," in *Proceedings of the IEEE/ACM International Conference on Hardware/Software Codesign and System Synthesis*, (New York, NY, USA), pp. 247–252, ACM, 2006.

[107] C. Raghunandan, K. S. Sainarayanan, and M. B. Srinivas, "Bus-encoding technique to reduce delay, power and simultaneous switching noise (SSN) in RLC interconnects," in *Proceedings of the Great Lakes Symposium on VLSI*, (New York, NY, USA), pp. 371–376, ACM, 2007.

[108] J. D. Z. Ma and L. He, "Formulae and applications of interconnect estimation considering shield insertion and net ordering," in *Proceedings of the International Conference on Computer-Aided Design*, (Piscataway, NJ, USA), pp. 327–332, IEEE Press, 2001.

[109] R. Arunachalam, E. Acar, and S. R. Nassif, "Optimal shielding/spacing metrics for low power design," in *Proceedings of the IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, (Washington, DC, USA), pp. 167–172, IEEE Computer Society, 2003.

[110] P. B. Morton and W. Dai, "An efficient sequential quadratic programming formulation of optimal wire spacing for cross-talk noise avoidance routing," in *Proceedings of the International Symposium on Physical Design*, (New York, NY, USA), pp. 22–28, ACM Press, 1999.

[111] P. B. Morton and W. Dai, "Crosstalk noise estimation for noise management," in *Proceedings of the Annual ACM/IEEE Design Automation Conference*, (New York, NY, USA), pp. 659–664, ACM, 2002.

[112] L. Macchiarulo, E. Macii, and M. Poncino, "Wire placement for crosstalk energy minimization in address buses," in *Proceedings of the Conference on Design, Automation and Test in Europe*, (Washington, DC, USA), pp. 158–162, IEEE Computer Society, 2002.

[113] K. Moiseev, A. Kolodny, and S. Wimer, "Power-delay optimization in VLSI microprocessors by wire spacing," *ACM Transactions on Design Automation of Electronic Systems*, vol. 14, no. 4, pp. 1–28, 2009.

[114] E. Macii, M. Poncino, and S. Salerno, "Combining wire swapping and spacing for low-power deep-submicron buses," in *Proceedings of the Great Lakes Symposium on VLSI*, (New York, NY, USA), pp. 198–202, ACM Press, 2003.

[115] S.-J. Ruan, E. Naroska, and U. Schwiegelshohn, "Simultaneous wire permutation, inversion, and spacing with genetic algorithm for energy-efficient bus design," in *Proceedings of the IEEE International Parallel and Distributed Processing Symposium*, (Washington, DC, USA), p. 233.1, IEEE Computer Society, 2005.

[116] S.-J. Ruan, E. Naroska, and C.-C. Chen, "Optimal partitioned fault-tolerant bus layout for reducing power in nanometer designs," in *Proceedings of the International Symposium on Physical Design*, (New York, NY, USA), pp. 114–119, ACM, 2006.

[117] Y. Nakagome, K. Itoh, M. Isoda, K. Takeuchi, and M. Aoki, "Sub-1V swing internal bus architecture for future low-power ULSI's," *IEEE Journal of Solid-State Circuits*, vol. 28, pp. 414–419, Apr 1993.

[118] R. Golshan and B. Haroun, "A novel reduced swing CMOS bus interface circuit for high speed low power VLSI systems," in *Proceedings of IEEE International Symposium on Circuits and Systems*, vol. 4, pp. 351–354, 1994.

[119] M. Hiraki, H. Kojima, H. Misawa, T. Akazawa, and Y. Hatano, "Data-dependent logic swing internal bus architecture for ultra low-power LSI's," *IEEE Journal of Solid-State Circuits*, vol. 30, pp. 397–402, Apr 1995.

[120] H. Yamauchi, H. Akamatsu, and T. Fujita, "An asymptotically zero power charge-recycling bus architecture for battery-operated ultrahigh data rate ULSI's," *IEEE Journal of Solid-State Circuits*, vol. 30, no. 4, pp. 423–431, 1995.

[121] H. Yamauchi and A. Matsuzawa, "A signal-swing suppressing strategy for power and layout area savings using time-multiplexed differential data-transfer scheme," *IEEE Journal of Solid-State Circuits*, vol. 31, pp. 1285–1294, Sep 1996.

[122] H. Zhang, V. George, and J. M. Rabaey, "Low-swing on-chip signaling techniques: effectiveness and robustness," *IEEE Transactions on VLSI Systems*, vol. 8, no. 3, pp. 264–272, 2000.

[123] T. N. Blalock and R. C. Jaeger, "A high-speed clamped bit-line current-mode sense amplifier," *IEEE Journal of Solid-State Circuits*, vol. 26, pp. 542–548, Apr 1991.

[124] M. Izumikawa and M. Yamashina, "A current direction sense technique for multiport SRAM's," *IEEE Journal of Solid-State Circuits*, vol. 31, pp. 546–551, Apr 1996.

[125] M. M. Khellah and M. I. Elmasry, "A low-power high-performance current-mode multiport SRAM," *IEEE Transactions on VLSI Systems*, vol. 9, no. 5, pp. 590–598, 2001.

[126] A. Maheshwari and W. Burleson, "Differential current-sensing for on-chip interconnects," *IEEE Transactions on VLSI Systems*, vol. 12, no. 12, pp. 1321–1329, 2004.

[127] N. Tzartzanis and W. W. Walker, "Differential current-mode sensing for efficient on-chip global signaling," *IEEE Journal of Solid-State Circuits*, vol. 40, no. 11, pp. 2141–2147, 2005.

[128] S. G. Younis and T. F. K. Jr., "Asymptotically zero energy split-level charge recovery logic," *Proceedings of International Workshop on Low Power Design*, pp. 177–182, 1994.

[129] Y. Moon and D. Jeong, "An efficient charge recovery logic circuit," *IEEE Journal of Solid-State Circuits*, vol. 31, no. 4, pp. 514–522, 1996.

[130] A. Vetuli, S. D. Pascoli, and L. M. Reyneri, "Positive feedback in adiabatic logic," *Electronics Letters*, vol. 32, pp. 1867–1869, Sep 1996.

[131] A. Kramer, J. S. Denker, B. Flower, and J. Moroney, "2nd order adiabatic computation with 2n-2p and 2n-2n2p logic circuits," in *Proceedings of the International Symposium on Low Power Electronics and Design*, (New York, NY, USA), pp. 191–196, ACM, 1995.

[132] M. M. Khellah and M. I. Elmasry, "Low-power design of high-capacitive CMOS circuits using a new charge sharing scheme," in *Proceedings of the IEEE Solid-State and Circuits Conference*, pp. 286–287, 1999.

[133] K.-Y. Khoo and J. Alan N. Wilson, "Charge recovery on a databus," in *Proceedings of the International Symposium on Low Power Electronics and Design*, (New York, NY, USA), pp. 185–189, ACM Press, 1995.

[134] P. P. Sotiriadis, T. Konstantakopoulos, and A. Chandrakasan, "Analysis and implementation of charge recycling for deep sub-micron buses," in *Proceedings of the International Symposium on Low Power Electronics and Design*, (New York, NY, USA), pp. 364–369, ACM Press, 2001.

[135] B. Bishop, V. Lyuboslavsky, N. Vijaykrishnan, and M. J. Irwin, "Design considerations for databus charge recovery," *IEEE Transactions on VLSI Systems*, vol. 9, no. 1, pp. 104–106, 2001.

[136] R. Downing, P. Gebler, and G. Katopis, "Decoupling capacitor effects on switching noise," *IEEE Transactions on Components, Hybrids, and Manufacturing Technology*, vol. 16, pp. 484–489, Aug 1993.

[137] H. Su, S. S. Sapatnekar, and S. R. Nassif, "Optimal decoupling capacitor sizing and placement for standard-cell layout designs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 22, no. 4, pp. 428–436, 2003.

[138] L. Wang and N. R. Shanbhag, "Energy-efficiency bounds for deep submicron VLSI systems in the presence of noise," *IEEE Transactions on VLSI Systems*, vol. 11, no. 2, pp. 254–269, 2003.

[139] M. Zhao, R. Panda, S. Sundareswaran, S. Yan, and Y. Fu, "A fast on-chip decoupling capacitance budgeting algorithm using macromodeling and linear programming," in *Proceedings of the Annual ACM/IEEE Design Automation Conference*, (New York, NY, USA), pp. 217–222, ACM, 2006.

[140] J. Fu, Z. Luo, X. Hong, Y. Cai, S. X.-D. Tan, and Z. Pan, "A fast decoupling capacitor budgeting algorithm for robust on-chip power delivery," in *Proceedings of Asia and South Pacific Design Automation Conference*, (Piscataway, NJ, USA), pp. 505–510, IEEE Press, 2004.

[141] H. Li, Z. Qi, S. X.-D. Tan, L. Wu, Y. Cai, and X. Hong, "Partitioning-based approach to fast on-chip decap budgeting and minimization," in *Proceedings of the Annual ACM/IEEE Design Automation Conference*, (New York, NY, USA), pp. 170–175, ACM, 2005.

[142] S. Zhao, K. Roy, and C.-K. Koh, "Decoupling capacitance allocation for power supply noise suppression," in *Proceedings of the International Symposium on Physical Design*, (New York, NY, USA), pp. 66–71, ACM, 2001.

[143] C.-P. Chen and M. D. F. Wong, "Optimal wire-sizing function with fringing capacitance consideration," in *Proceedings of the Annual ACM/IEEE Design Automation Conference*, (New York, NY, USA), pp. 604–607, ACM, 1997.

[144] S. H. Kulkarni, A. N. Srivastava, and D. Sylvester, "A new algorithm for improved vdd assignment in low power dual vdd systems," in *Proceedings of the International Symposium on Low Power Electronics and Design*, (New York, NY, USA), pp. 200–205, ACM, 2004.

[145] A. Srivastava, D. Sylvester, and D. Blaauw, "Power minimization using simultaneous gate sizing, dual-vdd and dual-vth assignment," in *Proceedings of the Annual ACM/IEEE Design Automation Conference*, (New York, NY, USA), pp. 783–787, ACM, 2004.

[146] D. G. Chinnery and K. Keutzer, "Linear programming for sizing, vth and vdd assignment," in *Proceedings of the International Symposium on Low Power Electronics and Design*, (New York, NY, USA), pp. 149–154, ACM, 2005.

[147] R. Rao, H. Deogun, D. Blaauw, and D. Sylvester, "Bus encoding for total power reduction using a leakage-aware buffer configuration," *IEEE Transactions on VLSI Systems*, vol. 13, pp. 1376 – 1383, dec. 2005.

[148] M. L. Mui, K. Banerjee, and A. Mehrotra, "A global interconnect optimization scheme for nanometer scale VLSI with implications for latency, bandwidth, and power dissipation," *IEEE Transactions on Electron Devices*, vol. 51, no. 2, pp. 195–203, 2004.

[149] B. J. Sheu, D. L. Scharfetter, P.-K. Ko, and M.-C. Jeng, "BSIM: Berkeley short-channel IGFET model for MOS transistors," *IEEE Journal of Solid-State Circuits*, vol. 22, pp. 558–566, Aug. 1987.

[150] J. R. Black, "Electromigration - a brief survey and some recent results," *IEEE Transactions on Electron Devices*, vol. 16, pp. 338–347, Apr 1969.

[151] D. Burger and T. M. Austin, "The SimpleScalar Tool Set, version 2.0," *ACM SIGARCH Computer Architecture News*, vol. 25, pp. 13–25, jun 1997.

[152] T. Sherwood, E. Perelman, G. Hamerly, and B. Calder, "Automatically characterizing large scale program behavior," in *Proceedings of Architectural Support for Programming Languages and Operating Systems*, (New York, NY, USA), pp. 45–57, ACM, 2002.

[153] E. Perelman, G. Hamerly, and B. Calder, "Picking statistically valid and early simulation points," in *Proceedings of the International Conference on Parallel Architectures and Compilation Techniques*, (Washington, DC, USA), pp. 244–255, IEEE Computer Society, 2003.

[154] SimpleScalar LLC. URL: `http://www.simplescalar.com`.

[155] Predictive Technology Model. URL: `http://ptm.asu.edu/`.

[156] M. Ghoneima and Y. Ismail, "Delayed line bus scheme: a low-power bus scheme for coupled on-chip buses," in *Proceedings of the International Symposium on Low Power Electronics and Design*, (New York, NY, USA), pp. 66–69, ACM, 2004.

[157] K. Moiseev, A. Kolodny, and S. Wimer, "Interconnect bundle sizing under discrete design rules," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 29, pp. 1650 –1654, oct. 2010.

[158] J. Cong, L. He, C.-K. Koh, and Z. Pan, "Interconnect sizing and spacing with consideration of coupling capacitance," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 20, pp. 1164–1169, Sep 2001.

[159] J.-A. He and H. Kobayashi, "Simultaneous wire sizing and wire spacing in post-layout performance optimization," in *Proceedings of Asia and South Pacific Design Automation Conference*, pp. 373 –378, feb 1998.

[160] S. Wimer, I. Koren, and I. Cederbaum, "Floorplans, planar graphs, and layouts," *IEEE Transactions on Circuits and Systems*, vol. 35, pp. 267 –278, mar 1988.

[161] N. Hanchate and N. Ranganathan, "A linear time algorithm for wire sizing with simultaneous optimization of interconnect delay and crosstalk noise," in *Proceedings of the International Conference on VLSI Design*, (Washington, DC, USA), pp. 283–290, IEEE Computer Society, 2006.

[162] J. A. Roy and I. L. Markov, "High-performance routing at the nanometer scale," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, pp. 1066 –1077, june 2008.

[163] C. Xu, L. Jiang, S. K. Kolluri, B. J. Rubin, A. Deutsch, H. Smith, and K. Banerjee, "Fast 3-d thermal analysis of complex interconnect structures using electrical modeling and simulation methodologies," in *Proceedings of the International Conference on Computer-Aided Design*, ICCAD '09, (New York, NY, USA), pp. 658–665, ACM, 2009.

[164] R.-B. Lin and C.-M. Tsai, "Theoretical analysis of bus-invert coding," *IEEE Transactions on VLSI Systems*, vol. 10, pp. 929 –934, dec. 2002.

[165] ILOG, Inc., "CPLEX 9.0 ." URL: http://www.ilog.com/products/cplex, 2003.

[166] D. Chen, E. Li, E. Rosenbaum, and S.-M. Kang, "Interconnect thermal modeling for accurate simulation of circuittiming and reliability," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 19, no. 2, pp. 197–205, 2000.

[167] L. Li, N. Vijaykrishnan, M. Kandemir, and M. J. Irwin, "A crosstalk aware interconnect with variable cycle transmission," in *Proceedings of the Conference on Design, Automation and Test in Europe*, (Washington, DC, USA), pp. 102–107, IEEE Computer Society, 2004.

[168] D. Ernst, N. S. Kim, S. Das, S. Pant, R. Rao, T. Pham, C. Ziesler, D. Blaauw, T. Austin, K. Flautner, and T. Mudge, "Razor: A low-power pipeline based on circuit-level timing speculation," in *Proceedings of the Annual ACM/IEEE International Symposium on Microarchitecture*, (Washington, DC, USA), pp. 7–18, IEEE Computer Society, 2003.

[169] The C Clustering Library. URL: bonsai.hgc.jp/~mdehoon/software/cluster/.

[170] M. H. Lipasti, B. R. Mestan, and E. Gunadi, "Physical register inlining," in *Proceedings of the Annual Symposium on Computer Architecture*, (Washington, DC, USA), pp. 325–335, IEEE Computer Society, 2004.

[171] G. Gonzalez, A. Cristal, D. Ortega, A. Veidenbaum, and M. Valero, "A content aware integer register file organization," *ACM SIGARCH Computer Architecture News*, vol. 32, no. 2, pp. 314–324, 2004.

[172] A. Sodani and G. S. Sohi, "An empirical analysis of instruction repetition," *ACM SIGOPS Operating Systems Review*, vol. 32, no. 5, pp. 35–45, 1998.

[173] P. Petrov and A. Orailoglu, "Low-power instruction bus encoding for embedded processors," *IEEE Transactions on VLSI Systems*, vol. 12, pp. 812–826, Aug. 2004.