A Report for

NSF Workshop on Micro/Nano Circuits and Systems Design and Design Automation: Challenges and Opportunities

Acknowledgements

This workshop was sponsored by the National Science Foundation CISE/CCF Division under grant CCF-2041598. We thank the NSF Program Director, Dr. Sankar Basu, for the support of this workshop and for providing valuable feedback to the drafts of this report. We are grateful to all the workshop speakers, panelists, and roundtable participants for their insightful and stimulating presentations and discussions. Many of the roundtable participants have directly contributed to the writing of this report. The workshop program and a complete list of the speakers, panelists and roundtable participants are provided in the appendix.

Gert Cauwenberghs, Jason Cong, X. Sharon Hu,
Pinaki Mazumder, Subhasish Mitra, and Wolfgang Porod
(Members of the workshop steering committee)

* For questions and comments, please contact X. Sharon Hu at shu@nd.edu.

Contents

Chapter 1

Executive Summary

Design and design automation of micro-/nano-circuits and systems are key enablers in advancing information technologies which have so profoundly changed all our lives over the past six decades. Research in design and design automation has created fundamental design principles and tools fueling the exponential growth of the number of transistors on integrated circuit chips over the past 60 years. Without innovations in micro/nano circuits and systems design and automation, billion-transistor chips that form the foundations of today's information age would not be a reality!

Moving forward, design and design automation of micro/nano circuits and systems face several challenges. On the one hand, business-as-usual approaches are plateauing. Traditional ways of improving silicon CMOS technologies or designing, verifying and testing integrated circuits and systems are approaching various limits: physical size, power and reliability limits as well as complexity limits. At the same time, our dependency on such systems continues to grow. On the other hand, the recent rise of machine learning and artificial intelligence applications coupled with recent advances in emerging nanotechnologies and NanoSystems creates tremendous opportunities to develop customized solutions for highly efficient and robust circuits and systems. By NanoSystems, we refer to systems across multiple scales – from integrated circuit chips all the way to very large-scale systems – built on nanotechnology foundations. The overall systems aspects, coupled with nanotechnologies that form the foundation, are emphasized.

To identify key challenges and future research directions in the field of Micro/Nano Circuits and Systems Design and Design Automation for the next decade, the NSF sponsored a (virtual) workshop on December 14-16, 2020. The workshop assembled over 200 researchers and leaders from academia, industry and government. The first two days of the workshop included 11 plenary talks and 3 panels covering the following five themes: (i) electronic design automation (EDA) tools and methodologies, (ii) foundational technologies and NanoSystems, (iii) artificial intelligence (AI) / machine learning (ML) / brain-inspired hardware design, and (iv) new application domains. On the last day, the workshop attendees were grouped into five roundtables on the above four themes as well as the fifth theme of physics-inspired hardware design.

This report summarizes the views expressed by the attendees of the five roundtable discussions as well as the speakers and the panelists. It first elucidates the background and rationale behind organizing the workshop. Detailed discussions on the current state and future directions/needs of research are presented on the five themes: Electronic Design Automation (Chapter 3), Foundational Technologies and NanoSystems (Chapter 4), ML/AI/Brain-Inspired Hardware Design (Chapter 5), Physics-Inspired Hardware Design (Chapter 6), and Application Domains beyond Circuits and Electronic Systems (Chapter 7). The current states and future trends/needs related to education and workforce training are summarized in Chapter 8.

The report concludes with the following recommendations to the NSF.

1 There is an immediate need for the NSF to help organize and coordinate federal/state-level awareness campaigns, at least at the levels of artificial intelligence, robotics and quantum computing campaigns, to emphasize

the critical importance and tremendous potential of hardware technologies and the increasingly crucial role of EDA.

2 The NSF, in partnership with stakeholders from government and industry, should seek to create a MOSIS-like infrastructure for technology access that provides fabrication and design support for NanoSystems built using industrially-offered advanced silicon CMOS and beyond-silicon CMOS technologies.

3 The NSF should help establish and support facilities for prototyping medium- to large-scale NanoSystems, beyond a few (1 to 1,000) stand-alone devices, by creating new hardware fabrication facilities or by expanding capabilities of exploratory fabs.

4 The NSF should help establish and support community-wide design infrastructures (preferably in the cloud) with both industrial-strength tools and open-source research EDA tools.

5 There is an immediate need for new research programs focusing on co-design for NanoSystems, connecting hardware circuits and architectures with applications on one end of the spectrum and foundational nanotechnologies on the other.

6 The NSF should pursue newly emerging challenges in EDA that are presented by massive complexity, reliability and security threats, and new 2.5D and 3D technologies,

7 The NSF should initiate new research programs on new EDA approaches to software productivity on heterogeneous hardware platforms for existing / new domains.

8 The NSF should initiate new research programs specifically focusing on co-design for AI/ML systems, and also conversely on the use of AI/ML techniques for EDA.

9 The NSF should pursue new approaches to computing and information processing that are based on alternative representations of information (beyond traditional voltages and currents) that might take advantage of physics-inspired dynamics.

10 In collaboration with the National Quantum Computing Initiative, the NSF should initiate a new research program on EDA for quantum computing, which supports efficient synthesis and compilation from applications in high-level programming specifications to the family of rapid expanding quantum computing devices, including future domain-specific quantum computing systems.

11 The NSF should pursue new application domains that might benefit from the well-established EDA approach in electronics that has proven to be so successful.

12 The NSF should must create ways to attract diverse high-school and undergraduate students to the critically important field of NanoSystems design and design automation, and foundational technologies to advance future computing. Special fellowships and internships should be developed.

A more detailed discussion of the recommendations to the NSF is presented in Chapter 9.

Immediately following the NSF workshop, on December 16, 2020, NSF also organized a (virtual) meeting to assess the academic needs for accessing semiconductor foundries and associated support infrastructure, and brainstorm ways to provide such access and support to US academic researchers. (We will refer to this event as the Foundry Meeting in this report.) More than 50 invited representatives from the government, academia, industry, and foundry service providers attended the Foundry Meeting. The report by the Foundry Meeting is available at https://nsfedaworkshop.nd.edu/assets/429148/nsf20_foundry_meeting_report.pdf.

Chapter 2

Background

The field of design and design automation of micro/nano circuits and systems promotes interdisciplinary research spanning computer science, computer engineering, and electrical engineering. This field has created key technologies without which it would be impossible to achieve advances in information processing which is an inseparable part of our everyday lives. For example, fundamental principles and tools created by this field have empowered Moore's Law scaling for over 50 years. Without design and design automation of circuits and systems, it would be impossible to create billion-transistor integrated circuits that form the foundations of today's information age!

Funding from the NSF has played an important role in the growth of this field over the past several decades. Recent NSF funding in design and design automation of micro/nano circuits and systems can be roughly grouped into the following areas:

- Traditional EDA of VLSI circuits and systems: This area includes high-level and logic synthesis, physical design and manufacturability, design verification, manufacturing testing, and ways to build robust systems to ensure reliable and secure operation. These topics represent the core areas of EDA. Challenges introduced by rising design complexity and advanced device technologies are addressed.
- Machine learning (ML) and artificial intelligence (AI) hardware, neuromorphic hardware and other domain-specific hardware: This is a rapidly growing area that addresses the urgent need for domain-specific accelerators, especially for low-power, edge applications (so-called *edge AI*). This research area emphasizes collaboration between computer scientists, hardware designers and EDA researchers.
- NanoSystems using beyond-silicon CMOS nanotechnologies: Beyond-silicon CMOS technologies include new logic devices, new memory technologies, new interconnects, and new integration technologies. To build NanoSystems that fully exploit unique features of these new technologies, collaborative research between device technologists, designers, EDA researchers and computer scientists is required.
- Architecture/System-level design: Advances in device technologies must be complemented by advances at the architecture and system levels. Examples include system-on-chip designs and design automation, network-on-chips, cross-layer power/reliability-aware design, hardware security, etc.
- Analog and mixed-signal designs: Digital systems have played a dominant role in information processing. However, analog and mixed-signal circuits are critical for digital systems to interact with the physical world. Applications include communication, sensing, actuation, etc. Recently, there is renewed interest in analog computing approaches to energy-efficient computing. .
- Quantum-like circuits and systems: While NSF funding in design and design automation of micro/nano circuits and systems are of broader scope than specific contributions to Quantum Computing (QC), it has supported new paradigms and technologies that bear resemblance to QC both from the standpoints of basic principles and applications targeted by them. Examples of supported projects include single flux quantum Logic and computing using probabilistic bits (p-bits). While these do not use q-bits as in QC, they may point to new directions for solving hard computational problems, i.e., integer factorization on the one hand and machine

learning on the other.

The field of design and design automation, like other technical fields, faces unique challenges. For example, **traditional** ways of improving silicon CMOS technologies or designing, verifying and testing integrated circuits and systems are approaching various limits such as physical size, power and reliability limits, as well as complexity limits. At the same time, our dependency on such systems continue to grow. This creates major research opportunities for new approaches beyond conventional paths.

Moreover, the recent rise of ML/AI applications, the recent trend towards computing at the edge, and recent progress in NanoSystems enabled by beyond-silicon CMOS technologies create new opportunities for customized solutions to designing electronic systems in contrast to general-purpose processors of the 20th century.

One imminent trend arising from the efforts to address the above challenges is the rise of blurred boundaries between traditionally separate fields. For example, more and more design and design automation researchers are collaborating with (and contributing to) adjacent fields such as ML/AI, cybersecurity, edge computing and device technologies. Such cross-disciplinary interactions raise several natural questions including: (i) what are the high-risk and high-return research topics? (ii) what other adjacent fields should EDA research aggressively seek collaborations with? (iii) where and how scientific findings should be disseminated in order to have the greatest impact given the interdisciplinary nature of EDA research? (iv) where should research funding come from and how should it be distributed to encourage more transformative research? etc. Answering these questions require forward thinking and planning.

In 2009, an NSF-sponsored workshop, titled “Electronic Design Automation—Past, Present, and Future”, was organized (see http://cadlab.cs.ucla.edu/nsf09/), and a final report was published [1], [2]. The workshop was extremely valuable in identifying emerging research trends and in providing funding guidelines. Since then, there has not been such a workshop in this field. A partial view of the progress of some aspects of where the field has moved appeared in [3].

The NSF Workshop on Micro/Nano Circuits and Systems design and Design Automation: Challenges and Opportunities, held on December 14–16, 2020, brought together more than 100 top researchers from the various design and design automation areas. The workshop included 11 plenary talks, 3 panels and 5 roundtables. The diverse group of researchers represented both academia and industry, and included rising stars as well as prominent leaders. To encourage more interactions among the attendees, poster sessions were also organized.

Bibliography

[1] R. Brayton and J. Cong, "NSF workshop on EDA: Past, present, and future (part 1)," *IEEE Design Test of Computers*, vol. 27, no. 2, pp. 68–74, 2010. DOI: 10.1109/MDT.2010.51.

[2] ——, "NSF workshop on EDA: Past, present, and future (part 2)," *IEEE Design Test of Computers*, vol. 27, no. 3, pp. 62–74, 2010. DOI: 10.1109/MDT.2010.70.

[3] S. Basu, R. E. Bryant, G. De Micheli, T. Theis, and L. Whitman, "Nonsilicon, non-von Neumann computing—part I [scanning the issue]," *Proceedings of the IEEE*, vol. 107, no. 1, pp. 11–18, 2019. DOI: 10.1109/JPROC.2018.2884780.

Chapter 3

Electronic Design Automation

Electronic Design Automation (EDA) tools and methodologies played a central role in managing the exponential increase of design complexity due to the Moore's Law scaling, and powered the electronic industry to realize cost-efficient digital revolution. However, it is significantly underinvested compared to other areas (such as computer networking or machine learning). In this section, we outline multiple areas of challenges and opportunities for EDA, including supports for scaling, programmability and adaptivity, safety and dependability, and more efficient large-scale optimization. We reason that EDA should go beyond hardware platform creation to help a much larger class of software programmers to cope with increasingly heterogeneous platforms. It can play a key role as a new technology enabler. We end with thoughts and recommendations of educating the next generation of EDA researchers and practitioners, and needs for long-term forward-looking funding support.

3.1 Scaling

Exponential scaling according to Moore's Law continued in the last decade, and will continue for at least another decade, despite numerous technical challenges. Beyond that, micro-/nano-circuits and systems are expected to get even more complex. As a result, the design complexity has also grown exponentially, demanding more efficient and scalable EDA technologies and tools for compute, memory, and interconnect design and optimization.

After more than a decade of research and development, 2.5D and 3D integration is finally adopted by the industry [1]. For example, the largest FPGA chip has over 90 billion transistors [2]. It is achieved by integrating multiple dies on a silicon interposer. Exciting progress is also being made on monolithic 3D ICs [3], [4], where devices can be densely packed and integrated without using large through-silicon vias (TSVs). Much more integrated research is needed for electrical, thermal and reliability analysis and optimization for future device integration technologies.

Interconnect continues to be a performance bottleneck as we scale, as the interconnect delay does not scale nearly as well as the device delay. A lot of progress was made in physical synthesis in the past two decades, which considered both logic optimization (such as gate sizing and fanout optimization) with physical design to address the timing closure problem. More novel design methodology and EDA solutions are needed to overcome the interconnect bottleneck at the microarchitecture level to consider the trade-off between clock frequency, throughput and latency. For example, it is worthwhile to revisit the concepts of systolic arrays [5], network-on-chips [6], or even asynchronous designs [7] and see if they can be applied to a wider class of designs and be applied in transparent fashion with the help of automated EDA tools. One interesting example is the recent work on automated interconnect pipeline insertion at the HLS level guided by floorplanning information, which can effectively tolerate the long latency of interconnects across multiple dies in a 2.5D integration, resulting in 2X clock frequency improvement [8]. More research is needed in this direction.

With device scaling, multiple levels of memory hierarchy are introduced to overcome the memory wall [9]. Automated tools are needed to optimize the data movement across different memory hierarchy and maintain the data consis-

tency. Such help is needed both at the time of platform design/creation, and at the time of programming/mapping a given platform, which will be discussed more in Chapter 4.

3.2 Programmability and Adaptivity

Fast evolution of integrated computing systems in the late 80s and 90s was followed by a revolution in wireless communications systems over the last two decades. This has driven the proliferation of wirelessly interconnected distributed computing systems for a range of real-time applications: humanoid and other robots [10]–[12], self-driving cars [13], [14], unmanned aerial vehicles [15], [16] and sensor networks [17]. Such real-time systems generally consist of on-board embedded computing platforms and sensors, wireless interfaces to other agents and edge computing nodes as well as access to the cloud facilitated by the edge computing devices. A key issue is the wide workload variability across the entire distributed computing platform due to the dynamic nature of the real-time applications cited. Wireless communications bandwidth demands for autonomous vehicles can range from a few Mbps to Gbps [18] depending on vehicle safety requirements and the state of traffic around the vehicle. For the same reason, compute demands on vehicle compute subsystems can range from 100s of GOPs to 10s of TOPs within fractions of a second, at worst (e.g. car driving at 70mph faced with an object emerging from the side of the highway onto the path of the vehicle with little warning). For these reasons, it makes sense to design both the wireless communications and compute hardware (typically consisting of multi-core CPUs and an array of GPU accelerators [19] at this point, but we expect customizable accelerators will also be widely used in the future) to dynamically adapt to the instantaneous computational workload demanded by the end application. In the above, the term “workload” is implied to have two attributes: throughput and accuracy. Taken together, these define the Quality-of-Service of the deployed application. Such an approach is contrasted against one of designing “static” hardware designed for the worst-case workload demand (highest accuracy, highest throughput). This would consume a lot of power all the time, increasing heat dissipation and reducing system reliability and safety. Novel EDA tools are needed to explore all aspects of such a workload-proportional design paradigm [20], [21] including methods that take into account computation vs. communication costs across distributed compute nodes. The underlying (adaptive) hardware architectures will be highly heterogeneous, consisting of processor cores, GPUs, various kinds of domain-specific accelerators (e.g. those for machine learning), sensors, mixed-signal components and wireless interfaces. Verification, post-silicon validation, design debug, manufacturing test and post-manufacture tuning of such systems will pose key challenges since existing design validation and testing techniques are not directed at high levels of heterogeneity and real-time adaptivity. The problem of testing real-time adaptive systems is an open problem and is complicated by the fact that circuit and system functionality must be ensured not just for circuits under nominal manufacturing process variations but also for circuits impacted by large workload variations and failures.

3.3 Safety and Dependability

Safety and dependability of the electronic system (e.g. the control of autonomous vehicles discussed in the preceding section)! rely heavily on the verification capability. Verification remains a significant challenge for modern EDA flows. From pre-silicon to post-silicon, all the way to system integration, verification takes a significant and growing amount of the product development cycle. Formal verification is a promising tool for reducing verification effort, if current bottlenecks can be overcome. Formal verification requires three steps: the creation of a mathematical model; the specification of properties; and a formal mathematical proof that the model adheres to the specified properties. Step 1 is relatively easy for digital designs, as these are already mathematical objects. However, often a model for an abstraction of the design is needed to enable scalables proofs. Obtaining such abstractions automatically from RTL designs is an open problem. When such abstractions are developed manually, it is critical that their soundness be

verified against the concrete RTL design. This is rarely done and remains a gap in existing design flows. For designs that include analog or even physical components, modeling is more challenging. Efforts such as [22] that provide frameworks for systematic digital models of analog components should be pursued to support easier and broader ability for formal modeling. Increasingly, SoC platforms have significant firmware components that are shipped as part of the platform. Their modeling, especially as part of the firmware-hardware co-verification, is important and largely unaddressed.

Step 2, specification, is difficult because it requires design knowledge and formal expertise. To mitigate this difficulty, several promising approaches should be pursued. First, a better integration of design and verification can drastically simplify both specification and verification. As an example, consider equivalence checking of an original and an optimized design. With no integration, the equivalence check is challenging and takes hours. With a bit of information generated by the optimizer, the equivalence check is trivial [23]. Second, approaches that bypass much of the need for formal specification are also very promising. In particular, a recent approach based on the idea of quick error detection (QED) leverages the idea of self-consistency [24] to check if a design is consistent with itself. Symbolic quick error detection (SQED) applies this idea to the verification of designs with instruction set architectures [24], and accelerator quick error detection (AQED) extends this to stand-alone accelerators [25]. Instruction-level abstraction (ILA) is a framework for providing an ISA-like abstraction for non-processor designs and can similarly be used to significantly reduce the effort required to specify properties for formal verification [26]. Increasingly verification extends to beyond functional verification to include security concerns. Specifying security properties is under addressed in existing tool flows.

Step 3 is to prove the properties. This requires a verification engine. Today's verification engines are model checkers built on top of powerful Boolean satisfiability (SAT) engines. Some recent efforts have also been made to build word-level model checkers that leverage the greater expressive power of satisfiability modulo theories (SMT) solvers [23], [27], [28]. Recent and current efforts in SAT, SMT, and model checking have led to dramatic performance improvements. Continued progress in these core automated reasoning techniques are crucial for the continued success and scalability of formal verification.

Despite the consistent progress in verification, it is far from sufficient to meet the rapid advancement of the electronic systems. Given the exponential scaling of circuit size, it is not an exaggeration to say that none of the large, industrial-scale designs have been completely formally verified. Most went through bounded checking before the verification time and resources were exhausted. Moreover, as we raise the level of design abstraction to behavior C/C++ languages or even domain-specific languages, such as TensorFlow or Halide, there is a growing semantic gap that makes verification much more challenging. Furthermore, the programmability and adaptivity presented in the preceding section present another set of the challenges and opportunities. Online checking and verification capabilities are needed as the underlying system evolves and adapts to the new operating requirements. Finally, supports of composability need to be formally defined and verified as we scale to build a system of systems. Related to the issue of composability is that of hierarchical verification. While hierarchical design methodologies have long been used to enable design scalability, there is no equivalent hierarchical verification methodology. Verification and validation tends to be largely done on flat designs, which is the primary limitation to its scalability.

3.4 EDA Beyond HW Platform Creation

While traditional EDA focused mainly on platform creation, for example, enabling the latest development of the latest CPUs and GPUs, we believe that EDA can play an even bigger role in programming the future heterogeneous platforms. Domain-specific hardware acceleration is one of the most promising approaches to combating the stagnation of single-thread performance scaling [29], [30]. Along this line, EDA has seen enormous success for decades in

addressing the challenges of designing and implementing highly complex heterogeneous computing devices that feature massive parallelization and extensive specialization. PetaFLOPS-on-a-chip is already in sight — the newly released NVIDIA A100 GPU (at 7nm node) offers a peak throughput of 600+ TeraFLOPS for executing workloads with structured sparsity. We hypothesize that efficient computing at ExaFLOPS-scale will soon become a reality by leveraging emerging technologies such as processing-in-memory, monolithic 3D, and wafer-scale integration.

Creating powerful hardware platforms is only the first step toward democratization of accelerator-rich computing. The overarching goal is to popularize acceleration for not only high-value but also long-tail applications that could benefit from special-purpose hardware. However, difficulties in programming accelerators have hindered their widespread adoption. Currently, an average programmer has to struggle with unfamiliar and complex tools/libraries only to run applications on a new accelerator, while only achieving a small fraction of the peak FLOPS available on the device.

There are a host of new opportunities for EDA to enable software-inclined developers to productively exploit accelerators for a novel domain. With the heterogeneous/non-uniform architectures, programmers must navigate a broad design and optimization space. This is compounded by the fact that accelerators rely on specialized hardware that currently requires programmers to manage many concerns explicitly in software. Moreover, accelerators often evolve rapidly in size, topology, and capability to adapt to changes in application demands and cost requirements [31]. Hence it is crucial to support quick (in days instead of many months) bring-up of software stacks that can adapt to a moving target "ISA". To address these challenges, it is necessary to rethink the abstraction and objectives of EDA algorithms and tools by providing domain experts with a better trade-off between design optimality, agility, and scalability. Notably, several promising efforts have recently emerged, aiming to build a reusable infrastructure to develop domain-specific abstractions for programs and target accelerators based on intermediate languages [32] and extensible IRs [33].

Most techniques used by the EDA community, such as synthesis, mapping, placement, routing, and verification, are crucial for developing efficient compilation of domain-specific accelerators, which typically use spatial architectures. One good example is the recent advancement in automated generation of systolic arrays onto FPGAs [34]. By extending EDA beyond platform creation, EDA can benefit not only tens of thousands of hardware designers, but also millions of software programmers and even potentially data scientists.

3.5 EDA as a Technology Enabler

Electronic Design Automation (EDA) will play a pivotal role as technology enabler [35]. We argue that its role will be much more important for the evolution of the electronic industry in the coming decade as compared to the last 40 years. Indeed, EDA is a well-established design methodology mainly for CMOS technology, albeit its rising complexity due to CMOS downscaling. Nowadays, we are witnessing the successful use of 3-dimensional device structures (from silicon FINFETS, to NanoWires and to NanoSheets), as well as the rise of carbon-based electronics (i.e., CNTFETs) [36] and 2-Dimensional electronics (e.g., MoS2, WS2 WSe2) [37]. These latter two technology families pave the way to device stacking, and thus to new ways of achieving 3-D system integration. The feasibility of these technologies (and combination/hybridization thereof) to realize competitive systems requires a new set of EDA tools, that may depart from the current commercial offering because of the widely-different nature in devices and interconnect. In a similar vein, cryo-circuits, ranging from cryo-CMOS to superconducting single flux quantum (SFQ) devices and to quantum computing (QC), require widely different suites of EDA tools for design. In particular, SFQ dataflows operate in pipeline mode, and require accurate path balancing as well as expressing logic in terms of majority functions [38]. QC requires special EDA tools to map quantum algorithms to reversible logic circuits and then to QC circuit libraries. In both cases, differences in abstractions and in the operation mode of the fundamental devices require the design of new EDA tools and flows. Moreover, without such new tools, including the ability to

use them open-source for their tuning to various circuit and technology styles, it is not possible to bring forward such technologies (e.g., SFQ, QC) to the level of assembling credible prototypes.

It is our belief that as CMOS scaling slows down and new technologies emerge, an unprecedented effort is required to design and realize new computing/communication systems that continuously provide higher performance (within an energy consumption envelope), because of the variety of technology substrates and realization styles. As an example, most emerging technologies will be used to realize co-processors, and their fate will emerge out of a strong competition in the coprocessor domain. EDA is necessary to support this competition with ammunition of even strengths for the various contenders [39]. The possibility of neglecting a competitive technology because of the lack of design tools may endanger the superiority in computing/communication power which is the strongest asset of our civilization.

Finally, it is our belief that EDA synthesis and verification tools have provided us with a compilation path from abstract languages and models to circuits. EDA has simplified much design, as even ultralarge-scale designs consist of a limited number of interconnected cells with regular physical layouts. Part of the complexity is shifted to the compilation process (EDA tools) running in software. This paradigm is mirrored by the shift of computing to streamlined architectures (i.e., RISC processors) where much of the computing complexity is moved to software by the compiler. Eventually, the use of QC can be seen as requesting a material to do computation by exploiting superposition and entanglement in an array of devices, whose control comes from a complex set of signals and operations that are the result of quantum compilation. Again, in this case, the compilation process exploits a simple set of hardware devices (that are fast and powerful) through the abstraction of computation and transfer of some of its complexity via compilation. In fact, a recent study revealed a rather surprising result that existing quantum compilation tools from academia or leading industry (such as Google and IBM) have an optimality gap as large as 45X [40], which implies significant opportunity for improving quantum compilation. In summary, hardware compilation is one of the highest achievements of EDA, and this is why EDA has a key importance as computing and communication move into new technologies and paradigms.

3.6 Optimization, Learning, and Scaling are the Keys to EDA's Future

The core of EDA is optimization. Optimization becomes more critical as system complexity grows, as scaling slows, and as system design becomes a multi-everything (physics, objectives, levels of hierarchy) optimization. Today, we live in an era of optimization, and will continue to do so for the foreseeable future. Rather, there is skyrocketing complexity of design optimization across a space of heterogeneity, 3D/4D, beyond-Moore, CMOS, von Neumann, security, resilience, and more. There is a catechism here. We want more comprehensive and accurate design space exploration to avoid leaving value on the table. We need design capability that enables us to correctly explore and choose across high-dimensional solution spaces. And design optimization capability is by definition stuck behind a Pareto of "faster, better, less resources – pick any two". Revisiting optimization, especially in the context of EDA, is utmost important.

Ultimately, EDA is successful if it enables the design process to scale. The key to this is being able to make higher-quality decisions earlier, and for more complex problem instances. When compute and schedule resources can be applied with greater effect, we enable the scaling of solution quality. This highlights several intertwined needs. (1) Scaling of the design process requires ability to "see ahead", i.e., to predict outcomes of downstream design optimization steps. So, we need to predict optimization outcomes. (When we can see even further ahead - e.g., from algorithm and microarchitecture to area and power in silicon - we call this pathfinding.) (Note that since flows and methodologies do not tolerate big loops, our predictions already must have ingrained pessimism - because optimism risks loops.) (2) We need to recover solution suboptimality that has been left on the table over the course of

decades, as the EDA industry and its research were driven by turnaround time requirements. (3) Measure and then improve. We need to quantitatively measure the sub-optimality gap of the EDA solutions and identify opportunities for improvement. (4) Interdisciplinary collaboration. We should reach out to other research communities, such as applied mathematics, statistics, operation research, and theoretical computer science to work together to constantly expand the EDA optimization toolbox. The rest of this section elaborates on these points.

(1) Predictable optimizations. Design space exploration should ideally explore the more powerful knobs more thoroughly. For example, microarchitecture and RTL have more leverage than physical floorplanning and power delivery strategy [41]. But budgeted efforts are skewed toward the latter stages of design. This is because optimizing and exploring in early stages has limited value – we can't predict the back end accurately enough, our decisions don't correlate to what can be closed and signed off – and so we just give up and beat on later implementation steps [42]. EDA must discover more predictable optimization methods, and be able to orchestrate these methods on emerging (parallel, distributed, AI) compute substrates with predictability of outcomes as the driving criterion. In many cases, prediction can be done in a constructive way. An example is the recent work in [8], which first constructs a coarse placement to identify long interconnects and then use the information to guide the high-level synthesis tool to add pipeline stages needed to achieve high-frequency designs.

(2) Optimization quality. As a research community and as an industry, we need to revisit how EDA is rooted in optimization. Optimization is the quest to do better. But we often face competing demands. EDA tools address high-stakes optimizations, every day. As researchers and developers, we are trained to formulate and attack these optimizations using integer linear programs (ILPs), multicommodity flows, assignment, dynamic programming, satisfiability, and so on [43]. But when facing the demand or criticism such as "We need the answer overnight", "The approach is impractical due to its runtime", or similar messages, the researchers and practitioners in both academia and industry have to cut a lot of corners. We have been conditioned to not want to beat problems to death as they are NP-hard after all [44]. Unfortunately, this has come at a cost. We are still nearly as ill-informed about suboptimality gaps for classical EDA optimizations, and the potential benefits of long-running, distributed CAD optimizations, as we were 20+ years ago [45], [46]. But in today's era of optimization, 1% matters especially when designs are used at a massive scale (e.g. in the cloud). Our research needs to address what CAD optimization in practice will need to look like in the foreseeable future: more intelligent and autonomous; deployed on distributed and cloud resources; optimizing expectations and Pareto frontiers rather than single expressions. This means serious investment in research on learning-enabled optimization (i.e., at a new nexus of ML and optimization), distributed and federated methods, the interface between discrete-combinatorial and continuous methods, and related directions [42], [47].

(3) Measure and then improve. We need to constantly measure the sub-optimality gap of each step of the EDA design flow and identify opportunities for improvement. Most steps of the design flow are NP-hard. Quantitative measurement is not trivial. A good example of the work done in the circuit placement community, which constructed placement examples with known optimals (PEKO) to measure the quality of leading academic and industrial tools [48]. It unveiled large optimality and spurred rapid advancement in circuit placement algorithms in the subsequent decade.

(4) Interdisciplinary collaboration. The intense interest in scalable optimization methods are shared by other research communities, such as applied mathematics, statistics, machine learning, operation research, and theoretical computer science to work together to constantly expand the EDA optimization toolbox. For example, one of the very early NSF IPAM (Institute for Pure and Applied Mathematics) workshops twenty years ago was dedicated to "Multilevel Optimization in VLSICAD" [49], [50], which led to successful application of multilevel optimization techniques to circuit placement [51]. Such interdisciplinary collaborations should be encouraged and supported by NSF. Culture changes (open sourcing, benchmarking, ...) can also boost investment in rapid adoption of the latest advancement in optimization [42], [52]. As we have seen in such rapidly-advancing fields as machine vision and NLP

(cf. Figure 1 of [53]), optimization comes with benchmarks, measured progress to reduce suboptimality, and tight translation paths between research and leading-edge practice – all of which will need to be part of a next chapter for EDA.

Bibliography

[1] Y. Xie, J. Cong, and S. Sapatnekar, "Three-dimensional integrated circuit design," *EDA, Design and Microarchitectures, New York: Springer*, vol. 20, pp. 194–196, 2010.

[2] I. K. Ganusov, M. A. Iyer, N. Cheng, and A. Meisler, "Agilex™ generation of Intel® FPGAs," in *2020 IEEE Hot Chips 32 Symposium (HCS)*, IEEE Computer Society, 2020, pp. 1–26.

[3] M. M. Shulaker, G. Hills, R. S. Park, R. T. Howe, K. Saraswat, H.-S. P. Wong, and S. Mitra, "Three-dimensional integration of nanotechnologies for computing and data storage on a single chip," *Nature*, vol. 547, no. 7661, pp. 74–78, 2017.

[4] T. Srimani, G. Hills, M. Bishop, C. Lau, P. Kanhaiya, R. Ho, A. Amer, M. Chao, A. Yu, A. Wright, *et al.*, "Heterogeneous integration of BEOL logic and memory in a commercial foundry: Multi-tier complementary carbon nanotube logic and resistive RAM at a 130 nm node," in *2020 IEEE Symposium on VLSI Technology*, IEEE, 2020, pp. 1–2.

[5] H. T. Kung and C. E. Leiserson, "Systolic arrays for VLSI.," Carnegie-Mellon University, Dept. Computer Science, Pittsburgh PA, Tech. Rep., 1978.

[6] A. Ivanov and G. De Micheli, "Guest editors' introduction: The network-on-chip paradigm in practice and research," *IEEE Design & Test of Computers*, vol. 22, no. 5, pp. 399–403, 2005.

[7] I. E. Sutherland, "Micropipelines," *Communications of the ACM*, vol. 32, no. 6, pp. 720–738, 1989.

[8] L. Guo, Y. Chi, J. Wang, J. Lau, W. Qiao, E. Ustun, Z. Zhang, and J. Cong, "AutoBridge: Coupling coarse-grained floorplanning and pipelining for high-frequency HLS design on multi-die FPGAs," 2021.

[9] W. A. Wulf and S. A. McKee, "Hitting the memory wall: Implications of the obvious," *ACM SIGARCH computer architecture news*, vol. 23, no. 1, pp. 20–24, 1995.

[10] M. Hirose and K. Ogawa, "Honda humanoid robots development," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 365, no. 1850, pp. 11–19, 2007.

[11] B. Adams, C. Breazeal, R. A. Brooks, and B. Scassellati, "Humanoid robots: A new kind of tool," *IEEE Intelligent Systems and Their Applications*, vol. 15, no. 4, pp. 25–31, 2000.

[12] E. Guizzo, "By leaps and bounds: An exclusive look at how Boston Dynamics is redefining robot agility," *IEEE Spectrum*, vol. 56, no. 12, pp. 34–39, 2019.

[13] R. Sell, M. Leier, A. Rassõlkin, and J.-P. Ernits, "Self-driving car ISEAUTO for research and education," in *2018 19th International Conference on Research and Education in Mechatronics (REM)*, IEEE, 2018, pp. 111–116.

[14] A. Eskandarian, *Handbook of intelligent vehicles*. Springer, 2012, vol. 2.

[15] S. Hayat, E. Yanmaz, and R. Muzaffar, "Survey on unmanned aerial vehicle networks for civil applications: A communications viewpoint," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 4, pp. 2624–2661, 2016.

[16] M. Mozaffari, W. Saad, M. Bennis, Y.-H. Nam, and M. Debbah, "A tutorial on UAVs for wireless networks: Applications, challenges, and open problems," *IEEE communications surveys & tutorials*, vol. 21, no. 3, pp. 2334–2360, 2019.

[17] N. Rathi, J. Saraswat, and P. P. Bhattacharya, "A review on routing protocols for application in wireless sensor networks," *arXiv preprint arXiv:1210.2940*, 2012.

[18] A. Moubayed, A. Shami, P. Heidari, A. Larabi, and R. Brunner, "Edge-enabled V2X service placement for intelligent transportation systems," *IEEE Transactions on Mobile Computing*, 2020.

[19] S. Liu, J. Tang, Z. Zhang, and J.-L. Gaudiot, "Computer architectures for autonomous driving," *Computer*, vol. 50, no. 8, pp. 18–25, 2017.

[20] R. Sen and D. A. Wood, "Energy-proportional computing: A new definition," *Computer*, vol. 50, no. 8, pp. 26–33, 2017.

[21] D. Banerjee, S. K. Devarakond, X. Wang, S. Sen, and A. Chatterjee, "Real-time use-aware adaptive rf transceiver systems for energy efficiency under BER constraints," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 34, no. 8, pp. 1209–1222, 2015.

[22] S. Herbst, B. C. Lim, and M. Horowitz, "Fast FPGA emulation of analog dynamics in digitally-driven systems," in *Proceedings of the International Conference on Computer-Aided Design*, 2018, pp. 1–8.

[23] C. Mattarei, M. Mann, C. Barrett, R. G. Daly, D. Huff, and P. Hanrahan, "CoSA: Integrated verification for agile hardware design," in *2018 Formal Methods in Computer Aided Design (FMCAD)*, IEEE, 2018, pp. 1–5.

[24] F. Lonsing, S. Mitra, and C. Barrett, "A theoretical framework for symbolic quick error detection," in *2020 Formal Methods in Computer Aided Design (FMCAD)*, IEEE, 2020, pp. 1–10.

[25] E. Singh, F. Lonsing, S. Chattopadhyay, M. Strange, P. Wei, X. Zhang, Y. Zhou, D. Chen, J. Cong, P. Raina, *et al.*, "A-QED verification of hardware accelerators," in *2020 57th ACM/IEEE Design Automation Conference (DAC)*, IEEE, 2020, pp. 1–6.

[26] B.-Y. Huang, H. Zhang, P. Subramanyan, Y. Vizel, A. Gupta, and S. Malik, "Instruction-level abstraction (ILA) a uniform specification for system-on-chip (SoC) verification," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 24, no. 1, pp. 1–24, 2018.

[27] A. Champion, A. Mebsout, C. Sticksel, and C. Tinelli, "The kind 2 model checker," in *International Conference on Computer Aided Verification*, Springer, 2016, pp. 510–517.

[28] A. Goel and K. Sakallah, "AVR: Abstractly verifying reachability," in *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, Springer, 2020, pp. 413–422.

[29] J. Cong, Z. Fang, M. Huang, P. Wei, D. Wu, and C. H. Yu, "Customizable computing—from single chip to datacenters," *Proceedings of the IEEE*, vol. 107, no. 1, pp. 185–203, 2018.

[30] W. J. Dally, Y. Turakhia, and S. Han, "Domain-specific hardware accelerators," *Communications of the ACM*, vol. 63, no. 7, pp. 48–57, 2020.

[31] E. Chung, J. Fowers, K. Ovtcharov, M. Papamichael, A. Caulfield, T. Massengill, M. Liu, D. Lo, S. Alkalay, M. Haselman, *et al.*, "Serving DNNs in real time at datacenter scale with project Brainwave," *iEEE Micro*, vol. 38, no. 2, pp. 8–20, 2018.

[32] Y.-H. Lai, Y. Chi, Y. Hu, J. Wang, C. H. Yu, Y. Zhou, J. Cong, and Z. Zhang, "HeteroCL: A multi-paradigm programming infrastructure for software-defined reconfigurable computing," in *Proc. 2019 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, 2019, pp. 242–251.

[33] C. Lattner, J. Pienaar, M. Amini, U. Bondhugula, R. Riddle, A. Cohen, T. Shpeisman, A. Davis, N. Vasilache, and O. Zinenko, "MLIR: A compiler infrastructure for the end of moore's law," *arXiv preprint arXiv:2002.11054*, 2020.

[34] J. Wang, L. Guo, and J. Cong, "AutoSA: A polyhedral compiler for high-performance systolic arrays on FPGA," in *Proceedings of the 2021 ACM/SIGDA international symposium on Field-programmable gate arrays*, 2021.

[35] L. Amarú, P.-E. Gaillardon, S. Mitra, and G. De Micheli, "New logic synthesis as nanotechnology enabler," *Proceedings of the IEEE*, vol. 103, no. 11, pp. 2168–2195, 2015.

[36] G. Hills, C. Lau, A. Wright, S. Fuller, M. D. Bishop, T. Srimani, P. Kanhaiya, R. Ho, A. Amer, Y. Stein, *et al.*, "Modern microprocessor built from complementary carbon nanotube transistors," *Nature*, vol. 572, no. 7771, pp. 595–602, 2019.

[37] G. V. Resta, A. Leonhardt, Y. Balaji, S. De Gendt, P.-E. Gaillardon, and G. De Micheli, "Devices and circuits using novel 2-D materials: A perspective for future VLSI systems," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 27, no. 7, pp. 1486–1503, 2019.

[38] O. Chen, R. Cai, Y. Wang, F. Ke, T. Yamae, R. Saito, N. Takeuchi, and N. Yoshikawa, "Adiabatic quantum-flux-parametron: Towards building extremely energy-efficient circuits and systems," *Scientific reports*, vol. 9, no. 1, pp. 1–10, 2019.

[39] E. Testa, M. Soeken, L. G. Amar, and G. De Micheli, "Logic synthesis for established and emerging computing," *Proceedings of the IEEE*, vol. 107, no. 1, pp. 165–184, 2018.

[40] B. Tan and J. Cong, "Optimality study of existing quantum computing layout synthesis tools," *IEEE Transactions on Computers*, 2020.

[41] D. Chapter, *International technology roadmap for semiconductors*, 2009. [Online]. Available: https://www.dropbox.com/sh/ia1jkem3v708hx1/AAB1fo1HrYIKClJNk0dB7YrCa?dl=0.

[42] A. B. Kahng, "Mlcad today and tomorrow: Learning, optimization and scaling," in *Proceedings of the 2020 ACM/IEEE Workshop on Machine Learning for CAD*, 2020, pp. 1–1.

[43] T. Lengauer, *Combinatorial algorithms for integrated circuit layout*. Springer Science & Business Media, 2012.

[44] R. S. Barr, B. L. Golden, J. P. Kelly, M. G. Resende, and W. R. Stewart, "Designing and reporting on computational experiments with heuristic methods," *Journal of heuristics*, vol. 1, no. 1, pp. 9–32, 1995.

[45] F. Brglez, "Design of experiments to evaluate CAD algorithms: Which improvements are due to improved heuristic and which are merely due to chance?" Citeseer, Tech. Rep., 1998.

[46] A. E. Caldwell, A. B. Kahng, A. A. Kennings, and I. L. Markov, "Hypergraph partitioning for VLSI CAD: Methodology for heuristic development, experimentation and reporting," in *Proceedings of the 36th annual ACM/IEEE Design Automation Conference*, 1999, pp. 349–354.

[47] A. B. Kahng, "Reducing time and effort in IC implementation: A roadmap of challenges and solutions," in *Proceedings of the 55th Annual Design Automation Conference*, 2018, pp. 1–6.

[48] C.-C. Chang, J. Cong, M. Romesis, and M. Xie, "Optimality and scalability study of existing placement algorithms," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 23, no. 4, pp. 537–549, 2004.

[49] [Online]. Available: http://www.ipam.ucla.edu/programs/workshops/multilevel-optimization-in-vlsicad/.

[50] J. J. Cong and J. R. Shinnerl, *Multilevel optimization in VLSICAD*. Springer Science & Business Media, 2013, vol. 14.

[51] T. Chan, J. Cong, and K. Sze, "Multilevel generalized force-directed method for circuit placement," in *Proceedings of the 2005 international symposium on physical design*, 2005, pp. 185–192.

[52] A. B. Kahng, "Looking into the mirror of open source," in *2019 IEEE/ACM Int. Conf. Computer-Aided Design (ICCAD)*, IEEE, 2019, pp. 1–8.

[53] S. Bianco, R. Cadene, L. Celona, and P. Napoletano, "Benchmark analysis of representative deep neural network architectures," *IEEE Access*, vol. 6, pp. 64 270–64 277, 2018.

Chapter 4

Foundational Technologies and NanoSystems

4.1 Background

Nanotechnologies are the foundations for building NanoSystems that are an indispensable part of all our lives. Advances in NanoSystems are critical for advances in diverse fields ranging from artificial intelligence (AI) to 5G/6G communication to quantum computing. Coming generations of abundant-data applications will process unprecedented amounts of loosely-structured data (such as streaming video, natural language, real-time sensor readings, contextual environments, or even brain signals) to overcome global grand challenges ([1]). For example, the performance demands of many AI workloads are doubling every 3-4 months – the GPT-3 by OpenAI requires 355 GPU-years to train [2]. Yet at this exact moment, when 21st-century applications are demanding the largest improvements in computing performance, **conventional** approaches to improving performance are stalling:

- Today, computing systems use large off-chip memory, and abundant-data applications are increasingly dominated by the time and energy shuttling data back-and-forth between computing and memory chips – the **memory wall**.
- Over the next decade, the conventional 2D (transistor) miniaturization will get increasingly difficult and will eventually stop – the **miniaturization wall**.
- The classical Dennard scaling has already stopped for over a decade – the **power wall**.

As a result, the computation demands of 21st-century applications far exceed the capabilities of today's systems, from energy-constrained embedded systems all the way to the cloud, and cannot be met by isolated "business as usual" improvements in technology, circuits and architectures. Fortunately, there are many research ideas at the level of foundational technologies (logic and memory devices, integration technologies, thermal solutions) and also at the level of NanoSystems that leverage the unique properties of such foundational technologies to create new and transformative architectures. The combination of foundational technologies and NanoSystems architectures promises to deliver unprecedented functionality, performance and energy efficiency of future computing systems. Without such continued advances, these next-generation applications cannot be realized. Thus, foundational technologies and NanoSystems are critical to the economic competitiveness, technology leadership, and national security of the United States (also articulated in [3]). This creates three urgent needs that the NSF should address:

- Create awareness of the criticality of foundational technologies and NanoSystems at the federal, state and local levels.
- Advance research in foundational technologies and NanoSystems and also emphasize translational research into industrial offerings.
- Attract a diverse set of students, educate them, and create a diverse next generation of workforce who will lead innovations in foundational technologies and NanoSystems.

4.2 What is Foundational Technology?

Foundational technologies can hold different meanings depending on the level of abstraction. Silicon and beyond-silicon materials such as 2D transition-metal dichalcogenides (TMDs) or 1D carbon nanotubes (CNTs) are "foundations" for nanodevices. Nanofabrication processes, such as layered transfer and low-temperature material integration, are also "foundations" for building integrated circuits. Similarly, device concepts such as negative capacitance field-effect transistors (NCFETs), 2D FETs based on TMDs, carbon nanotube FETs (CNFETs), NEM relays or various memory cell technologies (RRAM, MRAM, PCRAM, FeRAM) are "foundations" for circuits. Such circuits are "foundations" for architectures that, in turn, serve as "foundations" for the software stack (from system software to applications). Likewise, systems and applications are equally foundational because they directly influence the underlying technologies.

For this NSF CISE report, our major emphasis is on NanoSystems at the circuit and the architecture levels (built using nanodevices employing nanofabrication techniques) for 21st-century computing systems.

Such NanoSystems research is expected to focus on innovations at the circuits and architecture levels (and associated design methodologies) enabled by novel nanomaterial, nanofabrication and nanodevice concepts. From hardware demonstration standpoint, we expect that NanoSystems research will strive to build at least medium-scale circuits and systems to demonstrate the effectiveness, practicality and scalability of new concepts. This is in contrast to nanomaterial, nanofabrication and nanodevice concepts that are often demonstrated in hardware at the scale of a few transistors or a few memory cells (often < 1,000). Medium- and large-scale circuits and systems can be built on top of existing silicon infrastructure, e.g., new nanotechnologies integrated on top of silicon wafers to demonstrate interesting circuit- and system-level capabilities – a "sauce over pasta" (or "curry over rice") approach. Such a NanoSystems approach to foundational technology research will spur innovations in two ways:

- Medium to large-scale hardware demonstration targets will also spur fundamental innovations in materials, fabrication and devices (to demonstrate scalability with respect to speed, energy, area and robustness).
- Beyond traditional benchmarking of device ideas, it will help translate application-level needs into technology targets that in turn will generate new concepts at the nanomaterials, nanofabrication and nanodevice levels.

Examples of NanoSystems approaches to foundational technology research include Design-Technology-co-optimization (DTCO) and System-Technology-co-optimization (STCO) techniques [4], the N3XT approach to abundant-data computing [5], [6], Nano-electro-mechanical (NEM) relays and their optimization [7]–[9], ways to overcome imperfections and non-idealities in logic and memory technologies, computing inside memory arrays, and p-bits for probabilistic computing [10]. It is different from the bottom-up approach to semiconductor technology research that was the norm over the past several decades – mostly innovations around the (silicon) transistor and its miniaturization along a (pre-determined) path of various transistor-level area, speed and energy targets mostly driven by industry.

4.3 Co-Design Across Heterogeneous Technologies, Architectures and Applications

20th-century computing systems were largely dominated by the following factors:

- Silicon transistors.
- Classical technology scaling: Dennard scaling and transistor miniaturization, mostly using lithography. The 2000s and 2010s saw equivalent scaling (e.g., stress/strain, high-K dielectrics) and some DTCO.
- General-purpose processors (and the rise of accelerators in the 2010s).
- Emphasis on processor clock speed (followed by the focus on energy and throughput after the end of Dennard scaling).

21st-century computing systems already are (and will continue to be) dramatically different from the 20th-century ones. Domain-specific accelerators are already rising in the 21st century as the speed and energy benefits of classical technology scaling diminish. The diversity of applications, algorithms and accelerator hardware architectures is changing very rapidly. For example, more than 200 hardware accelerators for AI inference and training have been published over the past 3-4 years. Beyond AI, hardware accelerators for data analytics, graph processing, genomics and security are also growing. Similarly, an explosion of new concepts in foundational technologies and NanoSystems is also emerging: not only new transistor technologies, but also a wide variety of memory technologies, new sensing technologies, new interconnect technologies, new on-chip and inter-chip integration technologies, and new thermal technologies. Unlike innovations mostly around the silicon transistor and its miniaturization in the past, there is growing recognition about **combining** these wide variety of technologies in innovative ways to create new architectures optimized for various application domains. This creates the need for **co-design** across technology, architecture and application levels, different from the approach that dominated in the 20th century. Here is a sample of a few such co-design questions:

- Given a set of tasks from an application domain and a set of foundational technologies (e.g., for logic, memory and connectivity/integration), how do we jointly explore the space of algorithms, architectures and technologies that achieve the best possible application-level energy and execution times?
- How do we translate application-level needs (e.g., energy, throughput, latency) into technology-level targets (e.g., logic energy/speed/density, memory energy/speed/density, density of connections) and derive (new) technologies that meet these targets?
- Can circuit-, architecture- or application-level techniques overcome inherent imperfections, variations or reliability challenges associated with various foundational technologies?

Such unprecedented technology-architecture-application affinity creates unique opportunities for innovative EDA approaches as a technology enabler (Chapter 3) to optimize 21st-century computing systems not only with respect to classical metrics (energy, throughput, cost) but also emerging metrics (e.g., security, privacy, accuracy of results, robustness to manufacturing and environmental variations).

A wide variety of foundational technologies and NanoSystems are being pursued by researchers, too many to be covered exhaustively in this report. As discussed in [11], the three pillars for future computing systems include technologies for logic, memory and connectivity between memory and logic. To give the reader a flavor of the breadth of ideas related to these technologies, we provide an overview in this report. Several of these beyond-traditional silicon technologies have been implemented in industrial facilities (e.g., [12]–[14]).

- *Logic technologies:* Examples include 2D FETs, Carbon Nanotube FETs, CFET (Complementary Field-Effect Transistor), Coolcube 3D, FeFET (Ferroelectric FET), Forksheet, FETs operating at low temperatures, Nanosheet FETs, NCFET (Negative Capacitance FET), NEM relays, Oxide transistors, Reconfigurable FETs, Spin logic, Tunneling FETs.
- *Memory technologies:* Beyond traditional DRAM, SRAM and Flash, examples of new memory technologies include FeRAM (Ferroelectric RAM), MRAM (Magnetic RAM), PCRAM (Phase-Change RAM), RRAM (Resistive RAM or Oxide RAM), 2D Memory. Various aspects of these new memory technologies include single vs. multiple bits stored inside each memory cell, transistors vs. selectors as select devices inside arrays, and 1T1R vs. 1T*n*R structures. Recently, there has also been significant research emphasis on computation of certain functions inside memory itself (e.g., using Kirchhoff's current laws, spin logic).
- *Integration and interconnect technologies:* Examples include various flavors of 2.5D and 3D integration using bonding, Through-Silicon Vias (TSVs) and monolithic 3D integration, 2D materials as liners and diffusion barriers, and photonics for inter-chip and on-chip interconnects.

- *Thermal technologies:* Beyond traditional heatsinks, new thermal technologies are crucial for coming generations of three-dimensional ICs. Examples of advances in thermal technologies include thin film evaporation devices, capillary driven devices including advanced heat pipes and vapor chambers, single and two-phase microchannel cooling, thermal interconnects, 2D heat spreaders, phase change materials and Peltier coolers.
- *Physical computing technologies*: Discussed in Chapter 6.
- *Flexible electronics*: Flexible and high-performance devices based on 1D or 2D atomic materials that can offer diverse functionality (optics, electronics, sensing, etc), and are stackable, if needed, for 3D integration.

Beyond foundational technologies, it is critical to explore new NanoSystems architectures uniquely enabled by these nanotechnologies. Research in such NanoSystems is difficult for multiple reasons:

- Complexity of NanoSystems hardware demonstrations: There is no coordinated effort in the US to enable NanoSystems hardware demonstrations even at the medium scale. Existing academic infrastructure makes such hardware demonstrations very difficult by research groups.
- Lack of access: Very few of the foundational technologies discussed above are supported by industrial or large-scale research facilities. Very few research groups have access to technologies supported by such facilities in the US (in contrast to research groups in other countries, especially in China and Taiwan).
- Design enablement: Design tools are essential for enabling architects and designers to create new architectures using new foundational technologies. There is a lack of such design enablement, from PDKs and libraries to design tools and flows.
- Research culture: The research culture in many areas at the higher levels of the system stack promotes hardware technologies that are offered by mainstream foundries. There is little incentive to explore new hardware technologies not offered commercially.

Despite these challenges, there has been significant progress in the field of NanoSystems: hardware prototypes demonstrating the benefits of new technologies (compared to traditional approaches), EDA flows to create designs using new technologies, and architectural concepts exploiting new technologies. Examples of NanoSystems hardware prototypes include [8], [10], [15]–[37].

Such an era of innovations brings its own opportunities and challenges that have deep implications on research and development in foundational technologies and NanoSystems in the 21st century:

- Fundamental material/fabrication/device research vs. fabrication of hardware NanoSystems prototypes.
- Benchmarking at the device level vs. at the NanoSystems level.
- The scale, duration, and management of NSF-funded projects.

4.3.1 Fabrication and NanoSystems Hardware Prototypes

Academic research on foundational technologies in the 20th century (and the early part of the 21st century) mostly focused on fundamental materials, devices and fabrication aspects. With (most) innovations around the silicon transistor and its interconnection with other silicon transistors, academic research on materials and devices was largely decoupled from circuits and architectures. Thus, hardware demonstration at the level of a few devices was sufficient from an academic research standpoint. Approaches to integrate such devices and interconnects at a large scale with high yields were mostly an industrial activity. One prominent exception was Flash memory with deep coupling of architectural and system-level techniques with technology advances to overcome various sources of imperfections and non-idealities.

Moving forward, since we expect much deeper connections between foundational technologies (including integration and thermal technologies) and NanoSystems circuits and architectures (as discussed in Sec. 3.1 above),

NanoSystem-level integration, demonstration and measurements (energy, delay, robustness) become extremely important.

Thus, academic research for foundational technology and NanoSystems must advance across three axes in the 21st century:

- Performance: Examples include traditional metrics (such as energy, delay) as well as robustness metrics (such as variations, reliability, accuracy) both at the device and the system levels.
- Integration complexity: Examples include number of components integrated, number of 3D layers and the density of connections between various 3D layers, and heterogeneity of integrated technologies.
- Scalability: Examples include miniaturization of components on one hand and number of components integrated (overlapping with integration complexity) on the other.

Academic research, using existing academic infrastructure and NSF funding structures, can perhaps address one or two of these axes simultaneously – it may be difficult to address all the three axes simultaneously. Thus, it is extremely important to establish and support facilities in close coordination with academia for prototyping medium- to large-scale NanoSystems for experimentation purposes, beyond a few (1 to 1,000) stand-alone devices as is common today. This might take the shape of "exploratory" fabs where foundational technologies can be customized for realizing NanosSystem demonstrations (somewhat similar to the implementation of CNFETs, RRAM and monolithic 3D at the SkyWater Technology Foundry as part of DARPA's 3DSoC program). Such a lab-to-fab (or lab-to-new fab) effort is essential to compress the time between technology innovations and their adoption. Moreover, involvement of commercial foundries and fabrication facilities ensures that outputs from future programs have a path for eventual commercial adoption and therefore broad impact. However, technology implementations alone is inadequate – design enablement infrastructure to support design (e.g., through new architectural and EDA tools) is crucial.

4.3.2 Benchmarking

Similar to fabrication research, benchmarking of foundational technologies must also span the spectrum from the device level to the NanoSystems level. Thus, benchmarking of future foundational technologies and NanoSystems raise major questions. Some of these questions are relevant for traditional benchmarking as well and have generated less than satisfactory answers in the past with. Others are unique to heterogeneous foundational technologies and NanoSystems of the future. A few examples are given below:

- At what level of maturity should foundational technologies be benchmarked? At a conceptual level (without hardware measurements)? Or, only after hardware measurements are available? Or both?
- How do we benchmark integration and thermal technologies for example, beyond traditional transistor and memory technologies? Benchmarking of such technologies might inherently have to be connected to certain target NanoSystems.
- Certain foundational technology concepts (e.g., functionality-enhanced/reconfigurable devices, NEM relays, or physical computing approaches in Chapter 6) may be more impactful in an application context with varied application-dependent metrics (e.g., throughput, energy and accuracy of specific tasks). How do we create consistent benchmarking methodologies for varied application domains? How can we bridge the large gap between low-level device details to applications? How can new EDA approaches to unlock the benefits of the foundational technologies?
- How can non-idealities inherent in foundational technologies (drive strength, imperfections, variability, reliability, accuracy) be incorporated during benchmarking?
- How can foundational technologies be benchmarked with respect to non-traditional metrics (such as security, technology obfuscation, wearability) which may require detailed application-level understanding?

- Benchmarking often uses lots of assumptions resulting in apples-to-oranges comparisons between various foundational technologies. Selection of technology concepts used for benchmarking can also get biased (e.g., by sponsors with specific interests). How can we prevent misleading conclusions resulting from such situations?
- Beyond traditional benchmarking (which quantifies the effectiveness of a given technology concept with respect to certain metrics), how can we translate application needs (e.g., energy, throughput) into technology targets (e.g., energy and delay of logic gates and memory cells, connectivity between various components)? Is it possible to guide the creation of foundational technologies themselves from such derived targets? There is a critical need for fast and explainable frameworks for end-to-end joint co-exploration and co-optimization of technologies, architectures, and applications for these purposes.

4.3.3 Style of NSF Projects

The following points about foundational technologies and NanoSystems research are clear from the above discussions:

- End-to-end expertise, from devices and fabrication all the way to architectures and algorithms, is crucial.
- Given the breadth and depth required, longer-term timeframes (beyond traditional 3 years) are necessary.
- Continuity of research progress (driven by intermediate research goals) is essential along these longer timeframes.
- Involvement of prototyping facilities / exploratory foundries / industrial fabrication facilities is important for ongoing research to maintain compatibility with eventual commercial adoption.
- Given longer-term timeframes and breadth across multiple disciplines, the research must be able to adapt itself to new results.

Based on the observations, it is important for the NSF to consider new kinds of projects in addition to traditional funding typical of NSF projects. While the NSF does fund large-scale projects such as Expeditions in Computing or Engineering Research Centers, we envision such projects to be different in several ways:

- End-to-end expertise connecting foundational technologies to nanosystems architectures and EDA.
- Focused exploration backed by significant funding with the goal of nanosystem hardware demonstrations.
- Incentivize high risk system-level demonstrations (with room for failure) over short-term progress indicators alone (such as publications).
- 10-year timeframe with intermediate quantitative goals and frequent reviews (e.g., quarterly reviews, major yearly reviews with go/no-go criteria to track progress backed by major funding increments upon success).
- Involvement of major prototyping or industrial-scale facilities for the use of new customized technologies for demonstration of hardware prototypes
- Industry as funded entities to unlock collaborations with academic teams. Such a mechanism is expected to foster new levels of collaborations between industry and academia.

4.3.4 Moving Forward

- There is an immediate need for new programs around NanoSystems connecting hardware circuits and architectures with applications on one end of the spectrum and foundational nanotechnologies on the other – a co-design approach.
- It is of paramount importance to establish and support facilities for prototyping medium- to large-scale NanoSystems for experimentation purposes, beyond a few (1 to 1,000) stand-alone devices as is common today. This might take the shape of new hardware facilities or “exploratory” fabs where foundational technologies

can be customized for realizing NanosSystem demonstrations (vastly expanding the foundational technologies supported at scale, similar to the implementation of CNFETs, RRAM and monolithic 3D at the SkyWater Technology Foundry as part of DARPA's 3DSoC program). This is critical for lab-to-fab and lab-to-new fab translations, and compress the time between technology invention and its broad adoption.

- Beyond traditional focus on transistor, memory and sensing technologies, it is crucial to explore innovative integration approaches for realizing new NanoSystems. Prototyping facilities discussed above (in 2b) must support such integration activities as well.
- The NSF should facilitate access to beyond-silicon CMOS technologies offered by (industrial and research) fabs and enable new nanonystems research at the circuit and architecture levels. Currently, only select groups of researchers have access to such advanced technologies and advanced integration.
- In tandem with new foundational technologies and new NanoSystem hardware demonstrations, design enablement of NanoSystems is crucial. While today's advanced EDA tools will continue to support industrial offerings, there needs to be a major emphasis on new design and verification tools to address emerging nanotechnologies and the complexities of emerging NanoSystems. Conversely, a co-design approach to derive technology targets from application-level needs is also critical.
- Beyond traditional 3-year projects, NSF should consider significantly bigger projects at a 10-year scale (with intermediate milestones obviously). Such projects can be of two flavors: (i) existing Expedition or ERC type projects with large teams, and (ii) focused exploration backed by significant funding with the goal of demonstrating hardware prototypes in facilities (such as those discussed above in 2b). Such efforts can be coupled with innovative ways of promoting translational research in this domain.
- There is an urgent need for a better mode of collaboration between academia and industry in the domain of foundational technologies. One potential option can be NSF-funded programs jointly involving both academia and industry as performers together with translational components.

Bibliography

[1] [Online]. Available: http://www.engineeringchallenges.org/challenges.aspx.

[2] [Online]. Available: https://lambdalabs.com/blog/demystifying-gpt-3/.

[3] [Online]. Available: https://obamawhitehouse.archives.gov/sites/default/files/microsites/ostp/PCAST/pcast_ensuring_long-term_us_leadership_in_semiconductors.pdf.

[4] R. Kim *et al.*, "IMEC N7, N5 and beyond: DTCO, STCO and EUV insertion strategy to maintain affordable scaling trend," in *Design-Process-Technology Co-optimization for Manufacturability Conference*, 2018.

[5] M. Sabry Aly *et al.*, "Energy-efficient abundant-data computing: The N3XT 1,000X," *IEEE Computer, Special Issue on Rebooting Computing*, 2015.

[6] ——, "The N3XT approach to energy-efficient abundant-data computing," *Proceedings of the IEEE*, 2019.

[7] C. Chen *et al.*, "Efficient FPGAs using nanoelectromechanical relays," in *ACM International Symposium on FPGA*, 2010.

[8] ——, "Nano-electro-mechanical relays for FPGA routing: Experimental demonstration and a design technique," in *Design Automation and Test in Europe*, 2012.

[9] C. Qian *et al.*, "Energy-delay performance optimization of NEM logic relay," in *IEEE International Electron Devices Meeting*, 2015.

[10] W. Borders *et al.*, "Integer factorization using stochastic magnetic tunnel junctions," *Nature*, 2019.

[11] H.-S. Wong *et al.*, "A density metric for semiconductor technology," *Proceedings of the IEEE*, 2020.

[12] M. Bishop *et al.*, "Fabrication of carbon nanotube field-effect transistors in commercial silicon manufacturing facilities," *Nature Electronics*, 2020.

[13] Z. Krivokapic *et al.*, "14nm ferroelectric FinFET technology with steep subthreshold slope for ultra low power applications," in *IEEE International Electron Devices Meeting*, 2017.

[14] T. Srimani, G. Hills, M. Bishop, C. Lau, P. Kanhaiya, R. Ho, A. Amer, M. Chao, A. Yu, A. Wright, *et al.*, "Heterogeneous integration of BEOL logic and memory in a commercial foundry: Multi-tier complementary carbon nanotube logic and resistive RAM at a 130 nm node," in *2020 IEEE Symposium on VLSI Technology*, IEEE, 2020, pp. 1–2.

[15] M. Alamdar *et al.*, "Spin orbit torque domain wall-magnetic tunnel junction devices and circuits for in-memory and neuromorphic computing," *Bulletin of the American Physical Society*, 2021.

[16] F. Chen *et al.*, "Integrated circuit design with NEM relays," in *IEEE International Conference on Computer-Aided Design*, 2008.

[17] S. Dutta *et al.*, "Monolithic 3D integration of high endurance multi-bit ferroelectric FET for accelerating compute-in-memory," in *IEEE International Electron Devices Meeting*, 2020.

[18] E. Esmanhotto *et al.*, "High-density 3D monolithically stacked 1T1R multi-level-cell for neural networks," in *IEEE International Electron Devices Meeting*, 2020.

[19] M. Giordano *et al.*, "CHIMERA: A 0.92 TOPS, 2.2 TOPS/W edge AI accelerator with 2 MByte on-chip foundry resistive RAM for efficient training and inference," in *Symposium on VLSI Circuits*, 2021.

[20] G. Hills *et al.*, "Modern microprocessor built from complementary carbon nanotube transistors," *Nature*, 2019.

[21] P. Kanhaiya *et al.*, "Carbon nanotube-based CMOS SRAM: 1 kbit 6T SRAM arrays and 10T SRAM cells," *IEEE Trans. Electron Devices*, 2019.

[22] B. Le *et al.*, "Resistive RAM with multiple bits per cell: Array-level demonstration of 3 bits per cell," *IEEE Trans. Electron Devices*, 2019.

[23] K. MyNy *et al.*, "An 8-bit, 40-instructions-per-second organic microprocessor on plastic foil," *IEEE Journal Solid-State Circuits*, 2012.

[24] R. Radway *et al.*, "Illusion of large on-chip memory by networked computing chips for neural network inference," *Nature Electronics*, 2021.

[25] A. Raychowdhury *et al.*, "Computing with networks of oscillatory dynamical systems," *Proceedings of the IEEE*, 2018.

[26] M. Shulaker *et al.*, "Carbon nanotube computer," *Nature*, 2013.

[27] ——, "Three-dimensional integration of nanotechnologies for computing and data storage on a single chip," *Nature*, 2017.

[28] T. Srimani *et al.*, "Monolithic three-dimensional imaging system: Carbon nanotube computing circuitry integrated directly over silicon imager," in *Symp. VLSI Technology*, 2019.

[29] S. Wachter *et al.*, "A microprocessor based on a two-dimensional semiconductor," *Nature Communications*, 2017.

[30] W. Wan *et al.*, "A 74 TMACS/W CMOS-RRAM neurosynaptic core with dynamically reconfigurable dataflow and in-situ transposable weights for probabilistic graphical models," in *IEEE International Solid-State Circuits Conference*, 2020.

[31] Z. Wang *et al.*, "An all-weights-on-chip DNN accelerator in 22nm ULL featuring 24$\times$ 1 Mb eRRAM," in *Symposium on VLSI Circuits*, 2020.

[32] T. Wu *et al.*, "Brain-inspired computing exploiting carbon nanotube FETs and resistive RAM: Hyperdimensional computing case study," in *IEEE Intl. Solid-State Circuits Conf.*, 2018.

[33] ——, "A 43pJ/cycle non-volatile microcontroller with 4.7μs shutdown/wake-up integrating 2.3 bits-per-cell resistive RAM and resilience techniques," in *IEEE Intl. Solid-State Circuits Conf.*, 2019.

[34] J.-H. Yoon *et al.*, "A 40nm 64Kb 56.67TOPS/W read-disturb-tolerant compute-in-memory/digital RRAM macro with active-feedback-based read and in-situ write verification," in *IEEE Intl. Solid-State Circuits Conf.*, 2021.

[35] X. Xu *et al.*, "11 TOPS photonic convolutional accelerator for optical neural networks," *Nature*, 2021.

[36] C. Xue *et al.*, "A 22nm 2Mb ReRAM compute-in-memory macro with 121-28TOPS/W for multibit MAC computing for tiny AI edge devices," in *IEEE Intl. Solid-State Circuits Conf.*, 2020.

[37] S. Zanjani *et al.*, "3D integrated monolayer graphene–Si CMOS RF gas sensor platform," *Nature 2D Materials and Applications*, 2017.

Chapter 5

ML/AI/Brain-Inspired Hardware Design

5.1 Background

Ever since its inception, the field of computing has been enamored by the challenge of developing artificially intelligent systems [1], [2]. Taking inspiration from the biological brain, engineers have attempted to emulate its salient features at various levels of abstraction. This has resulted in a wide variety of neural and neurally-inspired algorithms, including today's Deep Learning (DL). Simultaneously, a bottom-up approach to emulating the brain's energy-efficiency and resilience led to the field of neuromorphic engineering, which uses the brain's information encoding principles to develop efficient hardware and algorithms [3]. Contrasting with the neuromorphic approach, highly abstracted models of the brain, for example those derived from dynamical systems research, or computation in extremely high-dimensional spaces [4], have shown promise from an algorithmic level.

Modern deep neural networks (DNNs) trace their origins back to the Perceptron [5], a highly abstracted model of biological neurons. Indeed, as the perceptron was being developed, so was the Mark I Perceptron machine, "intended as an experimental tool for the direct study of a limited class of perceptrons" [6]. Thus, although one might be tempted to consider algorithmic and hardware developments to have progressed independently, this has not historically been the case. The successful adoption and deployment of various algorithms in the field can be attributed to the serendipitous alignment between hardware design decisions and algorithmic and software design decisions together with external economic incentives [7].

DNNs, the dominant paradigm in contemporary research in machine learning/artificial intelligence (ML/AI) algorithms, heavily build upon multi-layer networks [8] and backpropagation [9]. Where previous algorithms were unable to scale, contemporary networks have shown remarkable success owing to gradient-based learning, the computational capabilities of GPUs, and the wide availability of large labeled datasets [9].

Given the wide-spread success of DNNs, one might question if it is beneficial to consider alternative algorithms. This can be answered with a resounding *Yes!* Alternative brain-inspired (BI) algorithms like spiking neural networks (SNNs), Hyperdimensional computing (HD), etc., offer different trade-offs in terms of accuracy, energy-efficiency, latency, and ease-of adaptation in the field. These differing strengths render them uniquely suited to tasks where constraints like latency, energy, or area are critical.

5.1.1 Applications of ML/AI/BI hardware design

The near-term applications of ML/AI/BI algorithms include: industrial control and automation, machine vision, natural language processing, strategic planning, electronic design automation, search, recommendation systems, time-series forecasting, and large-data analytics. However, their potential applications in the future are near-limitless. ML/AI/BI algorithms are poised to become an integral part of our lives, embedding these algorithms into ambient electronics will lead to way to pervasive intelligence systems. Such systems could enable a range of applications

including: computer generated graphics, privacy preserving analytics, automation and robotics, precision medicine, brain-computer interfaces, next-generation communication systems and spectrum sensing, manipulating multimedia and AR/VR. For such systems to become reality, ML/AI/BI systems must transition from centralized processing towards highly parallel, low-latency, closed-loop systems with on-board intelligence and learning. *In the longer-term, the next-generation ML/AI/BI algorithms will be enabling strategic technologies that are critical to the National Science Foundation's central mission*. However, unequal availability of computational resources could severely hamper the development of these technologies. Thus, enabling such computational resources in a sustainable fashion is imperative for US competitiveness in ML/AI/BI.

5.1.2 ML/AI/BI Workloads

The complex interactions between different ML/AI/BI algorithms and the underlying hardware can often lead to an order of magnitude difference in hardware performance. Thus, unique trade-offs offered by these different algorithms must be better understood to truly optimize the hardware.

Biophysical Foundations: Neuromorphic engineering has taken direct inspiration from the biophysics of computation in neural systems that operate with fundamental elements that are intrinsically noisy, sluggish, and unreliable and yet attain high levels of performance, agility, and reliability at the system level [10]. Despite an accelerating pace of research in this area, much work remains to be done, and a substantial gap remains in practical performance. This is due, in part, to the lack of a uniform set of standardized benchmarks and datasets consistent with those of the broader ML/AI community, calling for a concerted effort and allocation of resources in bringing neuromorphic and other BI approaches on par with the practical performance of other ML/AI approaches. Among other impediments that have slowed progress towards elevating computing to truly brain-like levels, it is necessary to stimulate a more effective dialogue between computer science and engineering on one end, and neuroscience and cognitive science on the other end.

ML and DNN Algorithms: ML and DNN algorithms are primarily composed of cascaded layers of linear algebra based computational kernels followed by an application of pointwise nonlinearities. This structure is repeated multiple times, often at different scales, making these networks an ideal target for energy-efficient hardware acceleration. The success of these algorithms combined with the heavy computational burden they impose on the hardware has resulted in widespread industry adoption of domain specific accelerators for ML. Indeed, multiple industrial research artifacts have outlined company specific workloads to detail design choices for different accelerators. Simultaneously, advances that use DNNs to advance fundamental science [11] has outlined the need for greater understanding of ML hardware and enable widespread adoption to promote further innovations.

Although recent work has predominantly focused on computer-vision applications of ML, other applications must be considered. It's critical that research into developing hardware for these algorithms carefully consider the use-case. Consider that large language models like transformers contain 10^{11} parameters and cannot be deployed on the edge. Thus, hardware-centric research into compressing such models for edge-deployment is required. Such edge-hardware, must be energy-efficient, compressible, and deliver results in real-time. At the same time, when served from a central cloud, specialized hardware for large language models must address a different set of constraints like a) model serving latency, b) Performance/Watt, c) precision in computing, and d) programmability and flexibility.

SNN Algorithms: SNN algorithms attempt to stay truer to their biological counterparts, using spike-encoding of neuron activity to transmit information between neurons. This encoding is often sparser than an equivalent DNN, a potential source for energy-efficiency for the underlying hardware. Additionally, spikes ensure multiplication-less computation, at the cost of complex temporal dynamics in the neurons and synapses.

Recent industrial efforts in the form of *IBM TrueNorth* [12] and *Intel Loihi* [13] have revitalized interest in this area, providing a platform for algorithm designers to experiment with. However, research prototypes have remained more ad-hoc. In part, this can be attributed to the lack of standardization of SNN models or datasets, which contrasts starkly with the MLPerf benchmarks in the ML field. Electronic hardware systems that emulate the brain operate under different constraints than their biological inspiration. Consequently, they must efficiently play to the strengths of modern silicon technology while learning from biology. Unable to implement arbitrary reconfigurable point-to-point wiring, silicon neuromorphic systems use time-division multiplexing emulate biology's concurrency in a protocol called Address Event Representation (AER). Although past SNN hardware designs predominantly relied on analog computation for energy-efficiency, technology scaling has rendered this advantage moot, consequently many recent examples of large-scale SNN hardware have been entirely digital [12]–[14].

Recent algorithms have demonstrated promising results using emerging, biologically plausible learning rules such as equilibrium propagation and predictive coding [15]. The hardware implications of using such learning rules has not been studied. In particular, work studying their robustness to analog variability and the ease of online learning on streaming data must be evaluated for edge-computing applications.

BI Algorithms: While DNN and SNN algorithms operate based on connectomics-like principles, there exist other algorithms that are still brain-inspired but do not subscribe to those principles. Hyperdimensional computing ascribes the brain's success to the high degree of parallelism and robustness observed in the brain and leverages these through hyperdimensionality and holographic representation of information. Hyperdimensional vectors are loosely defined as pseudorandom vectors with dimensionality exceeding thousands. Holographic representations (in this context) distribute the information content across the entire hyperdimensional vector. A judicious combination of associative memory and computation using these hypervectors provides very efficient means of implementing few-shot learning while simultaneously displaying fault-tolerance and low-latency operation. Another approach to brain-inspired computing uses nonlinear dynamics. Some well known models such as reservoir computing, Boltzmann machines, and Hopfield networks can be formulated as high-dimensional dynamical systems. Such networks often model an associative memory with the patterns stored as attractors of the network dynamics. In the classical Hopfield network, dynamics are implemented as optimization procedures that minimizes an energy function. Indeed, such algorithms offer a path towards bridging physics-based computing and SNNs or alternatively physics-based computing and DNNs.

Computing at the Edge and in the Cloud: Typically, ML algorithms are trained on datacenter-scale computing systems and deployed for inference in a variety of scenarios. Such models could be deployed across a variety of hardware including: a) datacenters running inference algorithms; b) battery-operated mobile-devices; c) home-assistant type devices; d) emerging applications like intelligent vehicles, and autonomous systems; and e) extremely power-constrained applications like medical-implants. Each of these application scenarios pose different constraints, eg., service latency is critical for both datacenter applications and for autonomous vehicles however, datacenter inferencing can typically operate with batched requests, while autonomous vehicles must operate on streaming-data with redundancy across frames/data-stream. Similarly, battery-operated mobile devices must operate with greater mind to energy-efficiency than voice-assistant like devices that are typically connected to an outlet. This leads to a complex set of trade-offs between energy, cost, latency, size, and capabilities.

Emerging applications such as robotics, intelligent wireless systems, autonomous vehicles, distributed smart sensors, and applications in the health and medical domain are not well suited to centralized cloud-based learning. Typical backpropagation based-learning requires that data be aggregated over time. These have been adapted to distributed edge-devices through federated learning systems. However, these emerging applications suffer from limited communication resources, real-time response requirements, as well as the need to rapidly adapt to new,

unlabeled data. This has resulted in a push towards augmenting edge-based computing systems with learning capabilities. Since biological brains demonstrate such capabilities, biologically inspired algorithms for learning and computing on streaming data offer a path towards achieving these goals.

5.2 Hardware Design Approaches for ML/AI/BI

The remarkable progress in ML/AI/BI algorithms over the past decade has resulted in a diverse set of proposed accelerators. Indeed, as ML/AI/BI algorithms have grown to demand ever increasing capabilities from the underlying hardware, hardware platforms have grown to meet these demands. Due to their ability to scale to larger designs, resilience, and ease of design in comparison to similarly complex analog systems, digital systems have come to dominate the accelerator landscape. However, analog computing solutions offer an elegant path towards extreme-energy efficiencies that rival that of the human brain. Going forward research must embrace a hetereogenous approach where both analog-computing and digital-computing might co-exist.

5.2.1 Digital Hardware Design

Modern ML accelerators have been limited to digital architectures primarily due to their scale, the requirements for high-speed and consequently their use of complex advanced process nodes, and need for programmability. Digital accelerators for ML workloads have had strong industry backing with Google's TPUs, Microsoft's Project Brainwave, and Sambanova's Datascale having used systolic architectures, field-programmable gate arrays (FPGAs), and coarse-grained reconfigurable arrays (CGRAs). For the most part, academic papers on ML accelerators have overwhelmingly focused on convolutional neural network based inference workloads. Academic research has generally focused on smaller-scale inference workloads with a focus on micro-architectural optimizations [16]. Although initial research focused on loop-blocking and data-flow analysis [17] to optimize the memory hierarchy, more recent work has addressed sparsity and increased model complexity due to parameter reduction [18]. The flexibility in computation and dataflow afforded by digital accelerators has resulted in a rich, multi-faceted, design-space [16], [19].

Digital hardware tailored to SNNs typically resemble their ML counterparts [20], composed of multiple cores computing neural and synaptic dynamics within an integration timestep. Since these architectures are dominated by the area and access energy for memories, research has focused on optimizing memory hierarchy and organization for storing synaptic parameters [21]. Indeed, neurons in such systems can be implemented efficiently without multipliers, and more complex neurons such as biophysically accurate models (Izhikevich or adaptive-exponential) are well suited to digital implementations. Hardware to support neuromorphic computing can generally be analyzed by looking at two complementary subsystems: a) the communication fabric that uses AER and b) the neural and synaptic computational subsystems. Recent research has tended to focus on the latter, computational, subsystem.

HD computing has shown promise in delivering similar accuracy as state-of-the-art ML algorithms while incurring significantly lower computational costs [22], [23]. At its core HD computing entails an encoding phase, where data is encoded into a hyperdimensional vector and a search phase which, based on the encoded query, searches for hypervectors within a high-dimensional auto-associative memory, returning the vector closest to the query. Tailored digital hardware and FPGAs implementing such algorithms have shown promise, additional research is required to employ HD computing in practice.

5.2.2 Analog Hardware Design

Analog and mixed-signal (AMS) computation continues to offer an elegant path towards extremely energy-efficient computation. On one hand, implementing energy-efficient computation for feature extraction and classification near or

at the sensor can have incredible advantages from both an energy-efficiency perspective and a privacy-preservation perspective [24]. On the other hand, AMS computation must overcome the challenges faced with analog design in advanced technology nodes: a) combating process-temperature variation, noise, and mismatch; b) reduced voltage headroom; and c) limited benefits from transistor scaling. However, AMS computation is naturally applicable to two scenarios: a) developing adaptive, energy-efficient sensing systems for feature extraction and analog feature-extraction circuits and b) embedding analog computation within memory subsystems. Crucially, it is currently unknown where digitization ought to occur in the processing chain and will likely need to be determined for different systems and tailored to individual algorithms.

Time and Frequency Domain Computing: Time/Frequency-domain computing [25] are particularly attractive for their ability to deliver analog-like computational efficiency using digital elements. By encoding information in pulse-widths or pulse-frequency such architectures can overcome many limitations of analog computing in advanced process nodes. Ideally, such systems would be able to operate at the energy-efficiency of analog charge/current domain computing while still benefiting from transistor density scaling and voltage-scaling. However, to-date practical implementations of time/frequency-domain computation still offer limited precision in computing and have not yet convincingly demonstrated an energy/precision advantage over charge/voltage-domain analog computation.

Compute-in-Memory: DNNs, SNNs, and some BI algorithms like HD computing are very memory-centric. Prior research showed that data-movement energy dominates the energy-dissipation of digital accelerators for AI algorithms [16]. Compute-in-Memory (CIM) architectures offer a promising avenue for reducing the data-movement associated with model parameters in DNNs and SNNs. CIM architectures store the model parameters in a memory array. Such architectures have been designed using either multiple memory technologies including: CMOS-based SRAM cells, emerging non-volatile memory elements like RRAM, FeFETs, STT-MRAM, CBRAM, and others. Practical implementations of compute-in-memory structures have been demonstrated with multiple SRAM [26] configurations as well as emerging memory technologies [27]. Consider an example RRAM-based CIM architecture with each memory element programmed to a conductance level proportional to a "synaptic weight". Applying a voltage proportional to the an input vector (via digital-to-analog converters or voltage drivers) results in a current through each element which is proportional to the product of the input and the stored conductance. Summing this current, converting it into a representative voltage, and digitizing it through an analog-to-digital converter (ADC) encodes the output of a matrix-vector multiplication.

The simultaneous increase in the density for storage and computation offered by CIM systems has driven significant research in this direction. The use of emerging NVM elements in particular, can improve memory density by $500\times$ when compared to distributed memory elements [28] and $6\times$ improvement compared to SRAM cells. Similar architectures leveraging other memory technologies together with content addressable memories can also be used for a) implementing the associative memory required for HD computing [29], and b) calculating the distance against a query vector [30]. Furthermore, it is often not sufficient to have a single memory array for most ML/AI/BI applications since (i) such applications require a large amount of memory and (ii) the size of a single CIM array is rather limited due to reliability, energy efficiency, etc. How to construct hierarchical CIM systems that can deliver desirable performance at the application level requires combined efforts from computer architects, circuit designers as well as EDA researchers.

5.3 Hardware Design Challenges for ML/AI/BI

Mapping Algorithms onto Hardware: Accelerators for ML/AI/BI algorithms have adopted a diverse set of base-computational and memory elements. For each such configuration of hardware, the workloads and computations being run on them can be partitioned and scheduled in many different ways. These aspects of data movement and

staging dominate the energy-efficiency and performance of different architectures [17]. Exploring the different ways in which data and operations for DNNs can be staged across the memory hierarchy and scheduled is an extremely important but difficult problem. Recent research into traversing this vast design space has shown great promise in improving hardware performance [31].

Recent work on developing SNN accelerators has focused on mapping SNNs reliably onto different analog and digital computing platforms [32]. Future research must address increased automation in such mapping efforts, including borrowing from the ML space for improved co-design efforts[31]. Hardware-aware schedulers and compilers to enhance SNN performance is a key missing element. Spike-scheduling forms a central component of such architectures, reorganizing the delivery of spikes between different cores, to minimize the memory accesses and latency [21]. Optimizing spike-scheduling remains under-explored, and future research into these avenues is important due to their ability to dramatically reduce the energy per time-step for SNN hardware. Additional avenues, examining SNNs using traditional computer architecture tools also shows potential for increased insight into hardware design for SNNs [33].

Improving Movement, Storage, and Scheduling of On-Chip Data: AI-like algorithms entail a lot of data-movement. In recent designs, this data-movement has typically been accomplished using single-cycle Network-on-Chips (NoCs) [16] or systolic-arrays [34]. SNN accelerators favor NoCs to receive and transmit spikes between computing units. SNNs often use an asynchronous NoC using the AER protocol [12], although *Loihi* [13] uses a more conventional switch-based synchronous NoC. While asynchronous spike-routing between source and destination has obvious advantages, the lack of support from EDA tools for asynchronous digital design has hampered the widespread use of asynchronous AER. Thus, efforts promoting such tools are an important avenue for future research. Despite early research showing promise in different spike-routing protocols and optimization, recent work has focused on the computation and memory aspects of SNN architectures. Future research comparing SNNs and their DNN counterparts must address on-chip communication networks.

Performance Analysis for Memory-Centric Computing: Just as with data-movement, ML/AI/BI workloads are memory-centric. It is important that the hardware designed for these workloads be compared fairly to encapsulate the complex interaction between algorithms and specialized hardware.

For wider adoption of HD computing, HD hardware and algorithms must address a central question: Given the same memory and compute resources, can HD computing still offer advantages at the application-level [35]? Answering this question will require: a) accurate energy and performance estimates for different hardware components; b) research in quantizing HD algorithms and comparing them to ML/SNN approaches; and c) improved understanding of the fault tolerance of HD computing and comparing that to ML/SNN approaches as well as encapsulating their effect on the underlying hardware [36]. This requires benchmarking and co-design efforts (eg., DARPA-PA-19-03-03) across the design layers encompassing algorithmic efforts all the way down to different technologies and materials. For example, certain forms of encoding and decoding might be significantly more energy-efficient given certain hardware primitives [29]. Particular care must be taken to ensure a fair comparison against ML/SNN algorithms and their respective hardware.

Similarly, this holds not just for different algorithms like HD, SNN, etc., but also different architectures. It is of paramount importance that future research evaluating CIM architectures analyze and evaluate performance fairly from contributions at the circuit-level to the system-level. Such evaluation must include: a) mapping and evaluating multiple layers to CIM architectures; b) accounting for the impact of using CIM on other on-chip components eg., external memory accesses [37]; c) benchmarking and evaluating the performance for a range of computational primitives; d) assess the performance trade-off between accuracy, latency, and energy-efficiency using analog computation approaches; and e) fairly evaluate iso-area/iso-accuracy/iso-technology digital architectures to truly determine com-

parative advantages of each approach. However, such comparisons and evaluations are not feasible given existing design-space exploration tools and evaluation frameworks. Thus, developing such tools will be invaluable to enabling further research in using analog CIM architectures.

Physical Design for Analog and CIM: Analog circuits, especially subthreshold circuits, have heavily influenced neuromorphic circuits and SNN hardware. However, the algorithmic requirements for long integration time-constants has resulted in a major challenge to such research. However, this challenge can be addressed by a judicious combination of digitally augmented analog computation. Emerging low-leakage devices such as NEMS-based switches and capacitors could address this long-standing limitations. Time-frequency domain computing has shown great promise in synthesizing analog computational elements by augmenting traditional digital design flows [38]. However, this approach is still in its infancy and further research is required to develop different computational-primitives.

Due to the memory-centric nature of ML/AI/BI algorithms, analog-computing based architectures must co-exist with memory in some fashion. Analog current/charge-based circuits are typically implemented as compute interspersed with distributed memory [27] or as computation implemented on the memory bitline [26]. Studies often call the dataflow resulting from such architectures: *weight stationary*. Despite this wide variety in memory-cell choice and their effect on computational density and latency, all these architectures implement low-energy inner-product computations with outputs accumulated over a current/charge bus. These outputs are typically digitized using an ADC, with the energy-cost of the conversion amortized by the parallelism in computation. Thus, activating fewer rows in parallel per ADC conversion results in more conversions needed for a given matrix-vector product. However, at the same time, activating more rows in parallel increases the dynamic-range requirements from the ADCs, in turn increasing the energy expended per conversion. This trade-off fundamentally resolves into one of area-efficiency, latency, and energy-efficiency. The complexity of these trade-offs is further compounded with the introduction of hybrid time/frequency-based ADCs/TDCs. Future research must examine these trade-offs to determine the practicality of these circuits.

Emerging architectures that use CIM-based computing have typically focused on ML algorithms. Remarkably, modern neuromorphic design has not focused on CIM-based architectures despite having roots in analog circuit design. In part, this can be attributed to the difficulties of implementing temporal dynamics with CIM arrays. Although there has been some recent work towards reconciling this disparity [39], many questions remain open and must be addressed to develop competitive SNN hardware. Research into alternative circuit primitives is critical to developing energy-efficient SNNs with temporal synaptic dynamics.

Low-Precision Computing: The energy-efficiency, latency, and accuracy of ML/AI/BI algorithms is greatly affected by the precision of computation. Indeed, training an image classification system with modern DL techniques can entail up to 10^{18} floating-point operations [40]. Decreasing the precision with which these operations are performed can greatly improve the energy-efficiency across all stages of data-access, movement, and computation, and yet only minimally impact overall accuracy of the computation during inference[1]. Due to these benefits, the use of lower-precision operations is expected to become standard practice in the near future, particularly for convolutional neural networks. Indeed, operating with lower-precision has played a central role in energy-efficiency improvements in recent digital ML accelerators [40]. Research that comprehensively explores this limit will be critical in the near-term future. As with other brain-inspired approaches, HD computing is very memory centric [42], [43] and consequently work on quantization of vectors and operating at low-precision shows considerable promise.

Circuits for AMS computation are more efficient at low bit precision computation than their digital counterparts, making AMS well suited to inference applications. As noted earlier, ADC energy-cost scales exponentially with

1. It is commonly assumed that high precision is required of the weights in a DNN in order for gradient-descent based learning to converge. However, careful gain scheduling in the weight updates can achieve high accuracy even with low-resolution mixed-signal weights [41].

bit-precision. In recent years, time-based analog-computing circuits have been leveraged for low-precision analog computation. There has also been some promising work on implementing time-domain SNNs. However, this hasn't been scaled beyond a single core and 3-bits of computing resolution. Future work must focus on seeing how such designs can scale to larger systems. Crucially, the limited resolution of such computation often limits algorithmic performance and future research must address these limitations.

Sparsity in Computation: Research into model compression and parameter reduction seeks to reduce the energy costs of accessing memories storing a) the model parameters and b) generated intermediate values [16]. This compression is often achieved by sparsifying the models through pruning low-impact weights [44]. The compression benefits offered by sparsity, can be critical to edge-operation. Consequently, edge-hardware accelerators must accommodate sparsity, and designers must develop architectures that can benefit from the reduction in memory accesses and computations offered by sparsity. However, sparsity greatly decreases parallelism and incurs additional costs in energy and latency due to pipeline stalls, synchronization overheads, and cache misses. Recently proposed digital designs have attempted to balance the performance gained from sparsity with the performance lost due to limited parallelism [18]. Further understanding and modeling of these trade-offs is crucial to enable truly on-chip intelligence. Sparsity can trade-off against robustness in analog computation, degrading the accuracy for CIM-based architectures. CIM architectures are well suited to implementing matrix-vector multiplications and appear to be fundamentally incompatible with sparse computations. This incompatibility must be resolved for CIM architectures to deliver competitive performance on current edge workloads.

Sparsity forms a critical aspect of SNN workloads which incur sparsity due to low spiking rates as well as model compression on SNNs [45]. As an example, since SNNs don't require multipliers and SNN models such as liquid state machines can be very sparse, the computational and data-movement demands for such a network are very different from recurrent ML models like long-short term memories (LSTMs). SNNs can offer unique benefits through different encoding schemes, such as through time-to-first-spike. Such an encoding scheme leads to progressively sparser activity in deeper layers of the networks, offering different trade-offs from conventional DNNs. Such sparsity can dramatically reduce the latency of data movement through the SNN model, leading to rapid classification. The complex data-dependent sparsity displayed by such models requires tailored hardware design. However, the interaction between the model, the hardware, and the inputs is not well understood and must be investigated in future research.

3D Integration: Thermal dissipation forms a major limitation of next-generation 3D integration technologies. Insufficient thermal dissipation, can be catastrophic for emerging memories, especially when applied to CIM architectures. The sparsity in computation and communication afforded by SNNs provides a viable path towards addressing this challenge. However, CIM systems are not optimized for sparsity in inputs or weights. Research into resolving this tension is critical in the development of future 3D-memory based CIM systems.

Near-Sensor Computing: The coming years will see a massive increase in data-generation. Communicating all this data to a centralized data-center to be remotely processed will be increasingly infeasible. Adaptive energy-efficient sensing and localized decision-making will be a must to address these future challenges. One of the first neuromorphic integrated circuits, designed by Misha Mahowald, was such a adaptive sensor. This sensor emulated the retina in an attempt to endow vision sensors with adaptation and feature extraction [3]. Over the years, the complex adaptation mechanisms of the first silicon retina have been streamlined to produce temporal-change detection based event-driven sensors such as the DVS, ATIS, DAVIS, as well as their audio counterparts. Simultaneously, the prevalence of deep learning, and the end of Dennard scaling has driven integrated circuit design towards tighter coupling between the processing and the sensing. This increased coupling, is evidenced by recent work demonstrating feature-extraction with hand-crafted features for image sensors and audio sensors, as well as

analog computation implementing some layers of a neural network prior to digitization [46]. These trends have also been prevalent in radio-frequency communication applications [24].

However, how such feature-extraction and noisy-processing affects down-stream computation and data-movement is not well understood. Future research must address this interaction. Future research must identify how computation and resources must be allocated at different stages of sensory-classification and decision-making. That is, what computation is better performed in the analog domain, at the ADC, and by the digital processing. Such research requires system-level [47] modeling and performance evaluation tools that encompass analog sensors and computation, which are currently lacking. Libraries of components could facilitate open-source style development of a range of hardware designs.

Model Adaptation: Recent work shows promising improvements in ML algorithm performance on various hardware platforms through model co-adaptation [48]. This form of co-adaptation has progressed greatly for digital accelerators, stemming from rapid design and estimation of accelerator performance [31]. Typically ML model search is characterized by three factors [49]: a) The search space, b) the search algorithm, and c) the evaluation strategy. The search space encompasses the space of all models that can be synthesized using this model-search strategy. The search algorithms are the workhorse that discover different model architectures within the search space. Finally, the evaluation strategy encapsulates the goal of this search, ie., the metrics that are being optimized for during the search. Most hardware-aware co-adaptation techniques include a metric for model size and number of operations within the evaluation strategy. However, recent work has shown an alternative approach to model co-adaptation [48]. This strategy decouples the search from the initial training for improved search time and flexibility in distilling a model to a diverse range of hardware for deployment [48]. Future research is required to make progress along all three fronts of evaluation strategy, search strategy, and an expansion of the model-space. Future applications of such model adaptation can be leveraged to tune models to emerging hardware architectures (CIM/Time-Domain/HD), which in turn can improve the energy-efficiency of different models on emerging architectures.

Developing Reconfigurable Hardware: Hardware-efficient ML algorithms have primarily been driven by model adaptation. Simultaneously tailoring the hardware to the ML model can further increase these benefits. In this vein, research that leverages high-level-synthesis and reconfigurable computing has delivered competitive results [50]. However, the overhead of enabling reconfigurability erodes many of the gains to be derived from accelerators. At the same time, the rapid pace of ML model development creates a need for flexibility and programmability in the underlying hardware. One promising approach to this problem provides limited reconfigurability in the hardware without sacrificing any performance. CGRAs are one such avenue for providing high-performance without sacrificing reconfigurability. CGRAs provide a generalized template for accelerator design, easing the hardware development and the software toolchain development [31].

Computing with Coupled Dynamical Systems: Computing with coupled-dynamical systems offers a promising alternative [51] to existing bottom-up neural network design strategies. Such strategies treat the neural network as a dynamical system with independent control over: a) the steady-state population dynamics and b) neuron activity like spiking statistics and transient population dynamics. Such a dynamical systems framing enables stable and interpretable population dynamics, irrespective of the network size and the type of neuronal connectivity (inhibitory or excitatory). One particular advantage of such a system, ends up being similar to the holographic representation in HD computing. That is, “memory” is distributed across the network rather than being stored in a distinct location. Such a unified dynamical system could also enable newer paradigms of computing to be included [52] as they are discovered, eg., the computational role of glial cells, dendrites, and astrocytes. Crucially, automated design flows could be leveraged to: a) search a large design-space for different equivalent dynamical system configurations, b) generate robust computational primitives that are tolerant to mismatch, noise, and analog variation induced

nonidealities.

5.4 Hybrid Models and Cross-Layer Design

It has been demonstrated time-and-again that cutting across abstractions tends to deliver out-sized performance gains. Indeed, one reason for biology's incredible energy-efficiency lies in the tight coupling between the "hardware" and "algorithm" in biological systems. In stark contrast, the ease of design afforded by developing abstractions has been the driving force behind the success of modern EDA. The walls raised by these abstractions have only been reinforced by the performance advantages offered by technology-scaling. However, in light of the halt of Dennard scaling, and the impending halt to miniaturization, we must rethink our designs and the walls raised by our abstractions.

However, rather than diminishing EDA's role, this paradigm shift might bring it into greater prominence. The role envisioned for EDA, in this case, becomes a cross-cutting facilitator that enables end-application users to quickly estimate their application's performance on various platforms. Similarly, from the other perspective, tools would be required to enable designers and device-engineers to estimate how their designs might operate under different workloads. The platonic idea of such a tool could enable true co-design where the hardware and application are co-adapted. This idealized tool would require advances in: automated neural architecture search, automated hardware-aware model adaptation, automated hardware design-space exploration, automated scheduling and mapping onto designed hardware, automated hardware-generation, and many others.

While the idealized tool described above is infeasible, this approach highlights potential EDA contributions. In particular, to facilitate rapid application-targeted design-space exploration. Additionally, for future applications that might require complex design-decisions, EDA tools must support the rapid adoption of new algorithms and information encoding, for example: spiking neural networks, hyperdimensional computing, computing with coupled dynamical systems; emerging application spaces: adaptation at the edge, implants and brain-computer interfaces, robotics and applications involving sensing, decision-making, and actuation; and new computational primitives such as analog building blocks and coupled-oscillators. These techniques all require that we rethink design from the ground-up.

Bibliography

[1] A. Turing, "I.—computing machinery and intelligence," *Mind*, vol. LIX, no. 236, pp. 433–460, Oct. 1950, ISSN: 0026-4423. DOI: 10.1093/mind/LIX.236.433.

[2] J. Von Neumann, *The computer and the brain*. Yale university press, 2012.

[3] C. A. Mead and M. A. Mahowald, "A silicon model of early visual processing," *Neural networks*, vol. 1, no. 1, pp. 91–97, 1988.

[4] P. Kanerva, "Hyperdimensional computing: An introduction to computing in distributed representation with high-dimensional random vectors," *Cognitive computation*, vol. 1, no. 2, pp. 139–159, 2009.

[5] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain," *Psychological review*, vol. 65, no. 6, p. 386, 1958.

[6] J. C. Hay, B. E. Lynch, and D. R. Smith, "Mark I Perceptron Operators' Manual," CORNELL AERONAUTICAL LAB INC BUFFALO NY, Tech. Rep., 1960.

[7] S. Hooker, *The hardware lottery*, 2020. [Online]. Available: https://arxiv.org/abs/2009.06489.

[8] M. Riesenhuber and T. Poggio, "Hierarchical models of object recognition in cortex," *Nature neuroscience*, vol. 2, no. 11, pp. 1019–1025, 1999.

[9] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural networks*, vol. 61, pp. 85–117, 2015.

[10] C. Mead, "Neuromorphic electronic systems," *Proceedings of IEEE*, no. 78, pp. 1629–1636, 1990. [Online]. Available: https://web.stanford.edu/group/brainsinsilicon/documents/MeadNeuroMorphElectro.pdf.

[11] C. Coelho, A. Kuusela, S. Li, H. Zhuang, T. Aarrestad, V. Loncar, J. Ngadiuba, M. Pierini, A. Pol, and S. Summers, "Automatic deep heterogeneous quantization of deep neural networks for ultra low-area, low-latency inference on the edge at particle colliders," *arXiv preprint arXiv:2006.10159*,

[12] P. A. Merolla, J. V. Arthur, R. Alvarez-Icaza, A. S. Cassidy, J. Sawada, F. Akopyan, B. L. Jackson, N. Imam, C. Guo, Y. Nakamura, *et al.*, "A million spiking-neuron integrated circuit with a scalable communication network and interface," *Science*, vol. 345, no. 6197, pp. 668–673, 2014.

[13] M. Davies, N. Srinivasa, *et al.*, "Loihi: A neuromorphic manycore processor with on-chip learning," *IEEE Micro*, vol. 38, no. 1, pp. 82–99, 2018. DOI: 10.1109/MM.2018.112130359.

[14] Q. Liu, O. Richter, C. Nielsen, S. Sheik, G. Indiveri, and N. Qiao, "Live demonstration: Face recognition on an ultra-low power event-driven convolutional neural network ASIC," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, Jun. 2019.

[15] B. Millidge, A. Tschantz, and C. L. Buckley, "Predictive coding approximates backprop along arbitrary computation graphs," *arXiv preprint arXiv:2006.04182*, 2020.

[16] Y.-H. Chen, T. Krishna, J. S. Emer, and V. Sze, "Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks," *IEEE journal of solid-state circuits*, vol. 52, no. 1, pp. 127–138, 2016.

[17] M. Gao, X. Yang, J. Pu, M. Horowitz, and C. Kozyrakis, "Tangram: Optimized coarse-grained dataflow for scalable NN accelerators," in *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*, 2019, pp. 807–820.

[18] A. Parashar *et al.*, "SCNN: An accelerator for compressed-sparse convolutional neural networks," pp. 27–40, Jun. 2017. DOI: 10.1145/3079856.3080254.

[19] N. Corporation, *The NVIDIA deep learning accelerator*, http://nvdla.org/index.html, Accessed: 2021-05-01.

[20] C. Frenkel, M. Lefebvre, J.-D. Legat, and D. Bol, "A 0.086-mm^2 12.7-pJ/SOP 64k-synapse 256-neuron online-learning digital spiking neuromorphic processor in 28-nm CMOS," *IEEE transactions on biomedical circuits and systems*, vol. 13, no. 1, pp. 145–158, 2018.

[21] C. J. Schaefer, P. Faley, E. O. Neftci, and S. Joshi, "Memory organization for energy-efficient learning and inference in digital neuromorphic accelerators," in *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*, IEEE, 2020, pp. 1–5.

[22] Z. Jin and H. Finkel, "Accelerating hyperdimensional classifier on multiple GPUs," in *2019 IEEE International Conference on Cluster Computing (CLUSTER)*, 2019, pp. 1–2. DOI: 10.1109/CLUSTER.2019.8891039.

[23] A. Rahimi, P. Kanerva, and J. M. Rabaey, "A robust and energy-efficient classifier using brain-inspired hyperdimensional computing," in *Proceedings of the 2016 International Symposium on Low Power Electronics and Design*, ser. ISLPED '16, San Francisco Airport, CA, USA: Association for Computing Machinery, 2016, pp. 64–69, ISBN: 9781450341851. DOI: 10.1145/2934583.2934624.

[24] S. Joshi, C. Kim, S. Ha, Y. M. Chi, and G. Cauwenberghs, "21.7 2pJ/MAC 14b 8×8 linear transform mixed-signal spatial filter in 65nm CMOS with 84dB interference suppression," in *2017 IEEE International Solid-State Circuits Conference (ISSCC)*, 2017, pp. 364–365. DOI: 10.1109/ISSCC.2017.7870412.

[25] Z. Chen and J. Gu, "A time-domain computing accelerated image recognition processor with efficient time encoding and non-linear logic operation," *IEEE Journal of Solid-State Circuits*, vol. 54, no. 11, pp. 3226–3237, 2019. DOI: 10.1109/JSSC.2018.2883394.

[26] H. Jia, M. Ozatay, Y. Tang, H. Valavi, R. Pathak, J. Lee, and N. Verma, "15.1 a programmable neural-network inference accelerator based on scalable in-memory computing," in *2021 IEEE International Solid-State Circuits Conference (ISSCC)*, IEEE, vol. 64, 2021, pp. 236–238.

[27] W. Wan, R. Kubendran, S. B. Eryilmaz, W. Zhang, Y. Liao, D. Wu, S. Deiss, B. Gao, P. Raina, S. Joshi, H. Wu, G. Cauwenberghs, and H. Wong, "33.1 a 74 TMACS/W CMOS-RRAM neurosynaptic core with dynamically reconfigurable dataflow and in-situ transposable weights for probabilistic graphical models," in *2020 IEEE International Solid-State Circuits Conference-(ISSCC)*, IEEE, 2020, pp. 498–500.

[28] D. Bankman, L. Yang, B. Moons, M. Verhelst, and B. Murmann, "An always-on 3.8μJ/86% CIFAR-10 mixed-signal binary cnn processor with all memory on chip in 28-nm cmos," *IEEE Journal of Solid-State Circuits*, vol. 54, no. 1, pp. 158–172, 2018.

[29] G. Karunaratne, M. Le Gallo, G. Cherubini, L. Benini, A. Rahimi, and A. Sebastian, "In-memory hyperdimensional computing," *Nature Electronics*, vol. 3, no. 6, pp. 327–337, 2020.

[30] K. Ni, X. Yin, A. F. Laguna, S. Joshi, S. Dünkel, M. Trentzsch, J. Müller, S. Beyer, M. Niemier, X. S. Hu, *et al.*, "Ferroelectric ternary content-addressable memory for one-shot learning," *Nature Electronics*, vol. 2, no. 11, pp. 521–529, 2019.

[31] R. Venkatesan, Y. S. Shao, M. Wang, J. Clemons, S. Dai, M. Fojtik, B. Keller, A. Klinefelter, N. Pinckney, P. Raina, *et al.*, "Magnet: A modular accelerator generator for neural networks," in *2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, IEEE, 2019, pp. 1–8.

[32] A. Balaji, S. Song, T. Titirsha, A. Das, J. Krichmar, N. Dutt, J. Shackleford, N. Kandasamy, and F. Catthoor, "NeuroXplorer 1.0: An extensible framework for architectural exploration with spiking neural networks," *arXiv preprint arXiv:2105.01795*, 2021.

[33] J.-J. Lee and P. Li, "Reconfigurable dataflow optimization for spatiotemporal spiking neural computation on systolic array accelerators," in *2020 IEEE 38th International Conference on Computer Design (ICCD)*, IEEE, 2020, pp. 57–64.

[34] N. P. Jouppi, C. Young, N. Patil, D. Patterson, *et al.*, "In-datacenter performance analysis of a tensor processing unit," *SIGARCH Comput. Archit. News*, vol. 45, no. 2, pp. 1–12, Jun. 2017, ISSN: 0163-5964. DOI: 10.1145/3140659.3080246.

[35] M. Hersche, E. M. Rella, A. Di Mauro, L. Benini, and A. Rahimi, "Integrating event-based dynamic vision sensors with sparse hyperdimensional computing: A low-power accelerator with online learning capability," in *Proceedings of the ACM/IEEE International Symposium on Low Power Electronics and Design*, ser. ISLPED '20, Boston, Massachusetts: Association for Computing Machinery, 2020, pp. 169–174, ISBN: 9781450370530. DOI: 10.1145/3370748.3406560.

[36] M. Hersche, S. Sangalli, L. Benini, and A. Rahimi, "Evolvable hyperdimensional computing: Unsupervised regeneration of associative memory to recover faulty components," in *2020 2nd IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS)*, IEEE, 2020, pp. 281–285.

[37] G. W. Burr, S. Lim, B. Murmann, R. Venkatesan, and M. Verhelst, "Fair and comprehensive benchmarking of machine learning processing chips," *IEEE Design and Test*, pp. 1–1, 2021. DOI: 10.1109/MDAT.2021.3063366.

[38] Z. Chen, H. Zhou, and J. Gu, "Digital compatible synthesis, placement and implementation of mixed-signal time-domain computing," in *2019 56th ACM/IEEE Design Automation Conference (DAC)*, 2019, pp. 1–6.

[39] B. Yan, Q. Yang, W.-H. Chen, K.-T. Chang, J.-W. Su, C.-H. Hsu, S.-H. Li, H.-Y. Lee, S.-S. Sheu, M.-S. Ho, *et al.*, "RRAM-based spiking nonvolatile computing-in-memory processing engine with precision-configurable in situ nonlinear activation," in *2019 Symposium on VLSI Technology*, IEEE, 2019, T86–T87.

[40] A. Agrawal, S. K. Lee, J. Silberman, M. Ziegler, M. Kang, S. Venkataramani, N. Cao, B. Fleischer, M. Guillorn, M. Cohen, *et al.*, "A 7nm 4-core AI chip with 25.6 TFLOPS hybrid FP8 training, 102.4 TOPS INT4 inference and workload-aware throttling," in *2021 IEEE International Solid-State Circuits Conference (ISSCC)*, IEEE, vol. 64, 2021, pp. 144–146.

[41] S. Joshi, C. Kim, C. Thomas, and G. Cauwenberghs, "Digitally adaptive high-fidelity analog array signal processing resilient to capacitive multiplying DAC inter-stage gain error," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 66 (11), pp. 4095–4107, 2019.

[42] S. Datta, R. A. G. Antonio, A. R. S. Ison, and J. M. Rabaey, "A programmable hyper-dimensional processor architecture for human-centric IoT," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 9, no. 3, pp. 439–452, 2019. DOI: 10.1109/JETCAS.2019.2935464.

[43] M. Imani, Z. Zou, S. Bosch, S. A. Rao, S. Salamat, V. Kumar, Y. Kim, and T. Rosing, "Revisiting hyperdimensional learning for FPGA and low-power architectures," in *2021 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, IEEE, pp. 221–234.

[44] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," *arXiv preprint arXiv:1510.00149*, 2015.

[45] E. O. Neftci, B. U. Pedroni, S. Joshi, M. Al-Shedivat, and G. Cauwenberghs, "Stochastic synapses enable efficient brain-inspired learning machines," *Frontiers in neuroscience*, vol. 10, p. 241, 2016.

[46] R. LiKamWa, Y. Hou, Y. Gao, M. Polansky, and L. Zhong, "RedEye: Analog ConvNet image sensor architecture for continuous mobile vision," in *2016 ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA)*, 2016, pp. 255–266. DOI: 10.1109/ISCA.2016.31.

[47] Y. N. Wu, J. S. Emer, and V. Sze, "Accelergy: An architecture-level energy estimation methodology for accelerator designs," in *2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2019, pp. 1–8. DOI: 10.1109/ICCAD45719.2019.8942149.

[48] H. Cai, C. Gan, T. Wang, Z. Zhang, and S. Han, "Once-for-all: Train one network and specialize it for efficient deployment," in *International Conference on Learning Representations*, 2019.

[49] T. Elsken, J. H. Metzen, F. Hutter, *et al.*, "Neural architecture search: A survey," *J. Mach. Learn. Res.*, vol. 20, no. 55, pp. 1–21, 2019.

[50] C. Coelho, A. Kuusela, S. Li, H. Zhuang, T. Aarrestad, V. Loncar, J. Ngadiuba, M. Pierini, A. Pol, and S. Summers, "Automatic deep heterogeneous quantization of deep neural networks for ultra low-area, low-latency inference on the edge at particle colliders," *arXiv preprint arXiv:2006.10159*,

[51] O. Chatterjee and S. Chakrabartty, "Decentralized global optimization based on a growth transform dynamical system model," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 12, pp. 6052–6061, 2018.

[52] ——, "Resonant machine learning based on complex growth transform dynamical systems," *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

Chapter 6

Physics-Inspired Hardware Design

6.1 Background

Physical Computing combines *physics* and *computation* in a complementary and synergistic fashion. On the one hand, one can exploit physics to efficiently perform a computational task, and, on the other hand, one can view computation as emerging from physics. Computing with Physics, encodes computation variables in physical quantities and the computation is performed using the physics of that particular medium. Physics as Computing, interprets physical state variables as computational quantities, and the time evolution of the physical system (according to the *Laws of Physics*) realized the computation.

While hardware implementation of all algorithms ultimately involve physical implementations, they do not necessarily mimic natural laws that our models (of physics) are meant to follow. For this to happen, it must be possible to map the problem at hand to the behavior of a system described by the natural laws of physics. The time evolution of the system should be describable not only by the (state) variables having physical interpretations, but also be constrained by fundamental conservation laws, e.g., energy, momentum, entropy etc. (admittedly, this requirement may limit the class of problems captured by this framework; nevertheless, improved performance may be achievable for an important and broad enough a class of problems, as is e.g., the case for other paradigms - AI/ML or brain inspired computing being examples). This latter requirement is often not meaningful (at least not directly taken into account) if other (nonphysical) variables are treated as primary variables in hardware implementation. More specifically, assuming that the problem under consideration can indeed be mapped to a system governed by the Laws of Physics, the set of state variables in an arbitrary hardware implementation, in principle, would have to have a mapping to the set of state variables for the associated model of physics. Depending on the structure of the computational problem under consideration, the map between these two sets of state variables may range from very simple to exceedingly complex, e.g., it could be an elementary one-to-one mapping (in which case a physical medium may be envisaged as a hardware substrate for computation), or be an isomorphism, or at an extreme could have disparate dimensions, and be highly complicated and nonlinear. As discussed later in this chapter, the potentially complex (nonlinear) nature of these mappings can give rise to behaviors of the hardware implementation significantly deviant from the behaviors required by the intended physics of the system (e.g., deviate from meaningful concepts of robustness, stability, dissipativity etc.), which may involve additional compensation.

Physical computing involves algorithms operating over the whole spectrum of variables in value and in time. The four types representing variables range from what is commonly known as analog to what is commonly known as digital and they are simply four different types of physical encoding of the variables. They are: continuous-value continuous-time (CVCT), continuous-value discrete-time (CVDT), discrete- value discrete-time (DVDT), and discrete-value continuous-time (DVCT) [1]. These include real-valued quantities, analog (electronic, magnetic, optic, spintronic, etc), quantum, neuromorphic, wave-based, phase / phasor, and cellular computation.

Digital computing computes over binary values and is modeled by a classical Turing Machine. Multivalue representation of information is also widely used today as for example in flash memory cells. A computation over physical quantities is not effectively modeled by an integer-value Turing machine, but could be modeled by a real-value Turing machine [2]–[5]. One expects an equivalence between each of these physical computing mechanisms where an application in one space can be transformed using finite polynomial resources to a solution in another space [3]. Physical computing often capatilizes on spatio-temporal wave-based processing, where these wave techniques allow for effective and efficient addressing the often limiting communications aspects of computing. whether it be interactions between qubits, coupled analog ODEs or PDEs, or coupled optical systems [3], [6]–[8]. These potentially continuous-time (or asynchronous) physical systems could enable solutions one cannot do on a digital computer, such as updating nodes in a graph problem in a synchronous fashion if there are conditional dependencies [9]–[11].

Mead's 1990 manifesto and much subsequent work have shown that *Physical Computing* techniques can offer great improvements in computational energy efficiency as well as area efficiency [12]. Analog computing in 100 nm nodes routinely has demonstrated $1000\times$ factors of computational energy efficiency over digital approaches [13], [14]. The cost of local computing drastically decreases from custom digital computation [15]. Today in the 10 nm nodes, mixed signal computing is still attractive and more efficient end-to-end with digital processing; the gap has however been diminished [16].

The round table on physics-inspired hardware design posed the following initial topic questions:

- What are promising physical domains and why? Perhaps a heterogenous mix of these.
- What are promising physical phenomena and why?
- Potential aspects of different physical computing approaches
- Similarities and differences between particular physical systems
- Do you see applications that would naturally map on the underlying physics?

6.2 Opportunities and Challenges

Several themes emerged during the RT discussion, which will be discussed below.

Conceptual Framework for Placing and Evaluating Physical Computing Approaches. Since there is a particularly wide variety of domains, concepts and terms used in Physical Computing, a framework for placing and making rational assessments — e.g., of credibility and potential value of a proposed approach — is especially necessary. Such a framework can be helpful for understanding the essence as well as the details of an approach, for comparing and contrasting different approaches, and for identifying clusters of related approaches. A suggested such framework is illustrated and explained below:

- Any proposed Physical Computing approach should first identify the operational concepts or principles it is based on. This should be expressed at the mathematical level, i.e., without bringing in details of implementation yet. As an illustration, a basic operational principle underlying [17]–[21] is to encode information (eg, logic bits, or spins) in the phases of oscillatory signals. [17], [20], [21] focus on making a finite state machine using this principle. [18] devises an Ising machine using this idea, with the additional principle that discrete Ising problems can be usefully embedded within a (continuous) mathematical representation of a network of coupled oscillators, i.e., the Kuramoto model. In [17], [18], the physical/mathematical phenomenon of oscillator injection locking is used to achieve bi-stable phase states, whereas in [19]–[21], the concept of parametric frequency division is used for the same purpose.
- Once operational principles have been clarified at the mathematical/conceptual level, the physical implementation of the principle can be explained, noting the physical domains involved and providing a system level

exposition. For example, [17], [18] can both be potentially implemented in diverse physical domains, from electronics to optics to synthetic biology; CMOS implementations in particular (using novel circuit structures) are immediately feasible and have many practical advantages. CMOS oscillator circuits, used to build higher level blocks corresponding to latches [17], or embedded in a network that mirrors the Ising problem [18], comprise the implementation at the system level. [19] uses parametric frequency division in the optical domain to devise binary spins, together with an electro-optical system to couple spins.

- Finally, as appropriate, details of low-level "device technologies" that enable/support the above conceptual levels, can be noted and their need/novelty/usefulness explained. For example, electronic implementations of [17], [18] system blocks can use MOS transistors in standard CMOS technologies for immediate practical realization, and can also leverage potential advances offered by future nanotechnologies. CIM employs degenerate optical parametric oscillators (DOPOs), laser pulses through optical fibers, Mach-Zehnder style electro-optical modulators, etc.

Computation vs. Communication. Although the computation can be highly efficient, communication of those results and the cost of moving data is expensive. Digital systems have the same communication issues, and yet, because of the higher relative costs of digital computation, the communication costs seem less severe. The current information-processing paradigm largely is based on the localized processing of information, with data shuttled to processing units from memory, which typically consists of a hierarchy of storage elements. This separation of data processing and data storage requires the efficient transfer of information through expensive channels.

Physical computing is nearly free where communication is the primary cost, including the communication to memories. *Physical Computing* architectures have different tradeoffs from classical digital approaches that must be addressed for competitive full system solutions. Computer architects [22] have been aware of the "memory wall" i.e. the challenges in speed an energy of moving data in the compute hierarchy [23]. These constraints have led to computing in memory architectures [24], [25] as well as mixed-signal [26]–[28]). Physical computation could use a range of distributed representations of information, including coherent waves where the processing of information is performed by physics-inspired propagation and interaction between waves through diffraction and interference.

Physical Computing Approaches to Discrete Optimization has recently Emerged as a Promising Direction. Recently, physical computing approaches have demonstrated considerable promise for solving difficult combinatorial optimization problems. Approaches include the D-Wave quantum annealer [29], coherent Ising machines [19], analog SAT solvers [30] and oscillator Ising machines [18]. Although the approaches used span diverse physical domains (quantum, optical, and electronic), a unifying feature is that discrete combinatorial problems are solved by embedding them into a continuous problem formulation. Future realizations may use other physical domains, such as (synthetic) biology. This is an important growth area in the long term.

Mapping the Structure of a Problem to an appropriate Physical System. Because of the range of computationally equivalent physical computing techniques, the designer must efficiently map the problem to the right physics-based solving medium. Finding the right representation enables exploiting the structure of the problem and its global information representation.

If one knows something about the structure of the problem, one can exploit the dynamics of the physical system to evolve towards the desired solution. For a search problem, for example, one might exploit noise in a smart way to push the search dynamics towards the solution. But if one does not know anything about the structure of the search space, one literally knows nothing.

The search space for computationally hard problems (NPC, NPH) is exponentially large, but only with $O(1)$ number of solutions. For this reason, searches that are similar to random walks in this space, will take exponentially long times to find a solution, they are "lost" most of the time. In order to make progress at solving some of these problems,

we need algorithms that can learn and exploit the structure of the search space. If one knows something about the structure of the problem, one can exploit the dynamics of the physical system to evolve it towards a solution. The challenge is that the trajectory of the search dynamics (for example, generated by the ODEs of a solver) is generated by a strongly non-linear dynamical system (only easy problems have a linear structure) and thus when computing the trajectory (by any hardware, including analog), small deviations (e.g., due to noise) will exponentially be magnified (positive Lyapunov exponents), effectively randomizing the search dynamics over long periods of time [31]. The key here is understanding and exploiting the interplay between accuracy of computation, rate of entropy generation by the system/device and time to solution (problem hardness).

Asynchronous Architectures. The RT agreed that there are problems one cannot do on a digital computer that one can do on an analog computer. For example, updating nodes in a graph problem cannot be done in a synchronous fashion if there are conditional dependencies. An asynchronous architecture can exploit randomness at the physical level, e.g. by randomly-flipping nanomagnets [32]–[34], where the probability that two magnets will flip at the same time essentially is nil. Such asynchronous architectures enable effective parallelism since while two nodes in the graph virtually never performs parallel updates, in the absence of a global clock, all nodes flip in parallel within a time constant of the physical substrate, e.g., the autocorrelation time of a nanomagnet. This leads to designs where increasing the number of nodes increases the throughput of the system, since each possible flip is a useful step in a larger computation [35]. *Physical Computing* appears to be a natural approach for asynchronous architectures.

Physical Computational Models. The digital computing paradigm is interwoven with computational models that rely fundamentally on discretization of problem-domains across real or complex multi-dimensional spaces. While domain-discretization offers certain advantages with respect to flexibility in defining the problem-space and choice of solution algorithms, it also poses certain unique and fundamental challenges to digital computing, in that digital (discretized) computational models may exhibit non-physical characteristics [36]; often, this may lead to non-trivial and costly challenges in digital computing [37].

For example, all physical systems are naturally causal [38]; however, digital computational models of physical systems may exhibit non-causal behavior [39], [40] that adversely affect computational stability and/or accuracy. As another example, passive physical systems are dissipative [41]; however, digital computational models of passive systems may generate energy [42] and lead to computational instability or inaccuracy. The enforcement of causality, passivity, and stability [43], in digital computational models of physical systems often face non-trivial obstacles that may include system pole/zero identification [44] or eigensystem perturbation [45]. Physical computation has the potential to offer an efficient modeling paradigm that intrinsically obeys physical laws. Other challenges imposed by digital computational models that would be naturally addressed through a physical computing paradigm involve the inherent approximation of infinite continuous sets via finite discrete sets, including **(a)** approximation of stochastic systems via series of deterministic systems [46]–[48], and **(b)** model-order-reduction to balance model size vs. model accuracy [49]–[51].

Multiple levels of Abstraction and Hierarchy of Scales. Physical Computing involves several basic levels of abstraction to address the wider hierarchy of spatial and temporal scales. Neurobiological and biological systems often compute using several orders of magnitude in both temporal and spatial scales. Physical computing solutions require computing processes at a range of large and fast scales, being responsive to current environmental changes, as well as a range of small and slow scales, which represent learning of the environment. Handling wide spatial and temporal scales maps well to the numerics of physical systems while being a difficult struggle (e.g. stiff ODEs) for digital computation [8], [10], [11], [52], [53]. Such hierarchically-structured dynamical systems, while common in nature, are largely unexplored in EDA.

Systems are needed that can create fluctuations at multiple scales in order to sample in space and time. These

fluctuations do not have to be quantum mechanical necessarily, they can be classical fluctuations. Further, if these kinds of fluctuations are fast compared to the dissipative processes that drive the adaptation of the system, then the system can rapidly sample a complex state and more slowly adapt itself. In this fashion, it can come to represent whatever it is that it is interacting with over a long period of time. Such a system would need processes at large and fast scales, which are responsive to whatever is going on right now, and it would need processes at small and slow scales, which represent learning of the environment. Instead of just large/small and fast/slow, one can envision a hierarchy of spatial and temporal scales [54], [55]. Indeed, machine learning techniques today implement these ideas in a limited way — the feed-forward pass of a deep neural net is a rapid, large-scale response to the current input, while the backward pass drives slow, small-scale adaptations that integrate the experience of many inputs and, thereby, improve the fast, large-scale response to similar inputs in the future.

Abstraction is essential for human design of these physical systems. The abstraction of physical systems, typical of digital computation (NAND/NOR, Multiply/Add, processors, algorithms), is possible for physical computing systems (e.g. [56]), although it requires understanding the core computational primitives of that substrate. These abstractions require formulating and modeling operational principles, implementation of these operational principles in a physical system, and addressing the technologies or devices being used.

Conceptual Bridges between Physical Computing Techniques. While visualizing a physical computing solution might be clearer with one technique, application constraints might require a different embodiment of the solution. Bridges must be built between these approaches; the conceptual framework noted above for placing approaches can help build such bridges. One example is state superposition, a phenomenon present in all physical systems when using their linear operating region (e.g. optical, analog, quantum) [3], [8], [53].

Different encodings allow for different implementation approaches. A complex number could be encoded as two real-valued physical quantities, or it could be encoded as the magnitude and phase of a sinusoidal signal. Different representations may have better interfaces to physical sensors and may enable improved implementations. In every application, there are tons of practical details that need to be solved. Interfacing one physical system to another physical system or interfacing a physical system to an integer-based digital system requires multiple engineering details as well as transformations / approximations between the domains.

Sometimes Noise can help Physical Computing. Nature teaches us that noise can be your friend. How does the cell that has incredibly noisy pico-Watt two- to three-bit precise analog variables, figure out when to divide and when not to, when to fight a virus, when to coordinate the immune system? It does that, because all the noisy analog variables collectively come up with one answer that matters. It is the final signal variable that matters, and not the intermediate signal variables that are noisy [4]–[8], [10], [11], [53], [57]–[60].

Noise can be your friend and it can be your enemy, and one needs both. For example, in [18], judicious addition of noise helps find better Ising minima, but too little or too much noise degrades results. Noise, or errors modeled as noise, will degrade the quality of a deterministic answer in any computation and can be the enemy of a designer. Several physical computing techniques are more robust to noise accumulation compared to digital systems, and yet accounting for noise is essential for all computations.

Noise often enables unbiased search decisions for good solutions to energy surface minimizations. A particular problem must utilize all knowledge of the system to reduce the energy landscape. If one just uses noise, then the system is just performing a random walk in an exponential large space and will be lost forever. These noise fluctuations should be created at multiple scales in order to sample in space and time. If these kinds of fluctuations are fast compared to the dissipative processes that drive the adaptation of the system, then the system can rapidly sample a complex state and more slowly adapt itself.

Phase and Amplitude. The RT discussed that there might be significant benefits in utilizing phase, and it was envisioned that future data-processing systems might be based on phase as much as on amplitude. Current information-processing systems are based on amplitude, and if one were to add phase to amplitude, one could build fundamentally more powerful devices. It appears that there is big room here for improvement here. Examples of already-demonstrated, practical systems based on phase include [17]–[19], as already noted earlier. Perhaps most interestingly, by including phase one then can then utilize state superposition. It is commonly believed that basing computation on states with superposition is an attribute only for quantum computers, and one has to have quantum computers in order to gain the benefits of superposition for algorithms. However, one can use classical wave superposition in classical devices, where no "quantum" is required. The challenge is how to use phase, in addition to amplitude, to build more functional devices that utilize superposition of states for information processing, and this is a big and largely unexplored area with significant promise.

Coherent Ising Machines (CIM) are an example of this approach [61]. Phase and amplitude can be used in wave-based [62] or coupled-oscillator-based [63] approaches to computing. Spin-wave based realization of optical computing primitives have been pursued [64]. Magnonic interferometric devices have been shown to be capable of prime factorization [65] and multi-valued logic operation [66]. Computing with networks of oscillatory dynamical systems has been discussed [67], [68] and specific physical implementations have been proposed [69] for specific applications [70]. A review and perspective of coupled oscillators for computing has been published recently [63].

6.3 Physical System Design requires an EDA Community Ecosystem

An Electronic Design and Automation (EDA) community ecosystem doing *Physical Computing* is needed to accelerate development to commercial timescales. The development of tools and computational framework for *Physical Computing* applications becomes essential for its long-term development. The infrastructure around large-scale Field Programmable Analog Arrays (FPAA) provide a good example of the efforts required [71], [72]. Another example is CAD tools for oscillator-based systems [73], which were instrumental in designing the phase-based computing systems [17], [18].

This ecosystem means *education* of the current and next generation of researchers must happen to empower these *Physical Computing* approaches. Given the interdisciplinary nature of these efforts, as well as the need to develop the larger computing stacks for these technologies, we need to raise up tall-thin people characteristic of the early digital VLSI development[74] as well as experts in the various subdomains.

One might imagine requiring education spanning fundamental physics, circuits, and system design levels, as well as application space knowledge. There is a need to create an ecosystem of people who are trained beyond doing ones and zeros and software on only those ones and zeros. We envision an ecosystem where individuals are trained in analog circuits, nonlinear dynamics, computational theory, physics, and related areas, as well as those who appreciate their importance. Hands-on and interactive workshops, such as the NSF supported *Telluride Neuromorphic Workshop*, would enable these educational opportunities, both for current students and for practitioners in these fields.

Bibliography

[1] P. M. Furth and A. G. Andreou, "Comparing the bit-energy of continuous and discrete signal representations," *4th Workshop on Physics and Computation (PhysComp)*, pp. 127–133, 1996.

[2] J. Hasler, "Opportunities in physical computing driven by analog realization," in *First IEEE International Conference on Rebooting Computing*, IEEE, Oct. 2016.

[3] J. Hasler and E. Black, "Physical computing: Unifying real number computation to enable energy efficient computing," *Journal of Low Power Electronics and Applications*, vol. 11.2, no. 14, 2021.

[4] R. Sarpeshkar and M. O'Halloran, "Scalable hybrid computation with spikes," *Neural Computation*, vol. 14, no. 9, pp. 2003–2024, Sep. 2002.

[5] R. Sarpeshkar, "Analog versus digital: Extrapolating from electronics to neurobiology," *Neural Computation*, vol. 10, pp. 1601–1638, 1998.

[6] J. J. Teo, S. S. Woo, and R. Sarpeshkar, "Synthetic biology: A unifying view and review using analog circuits," *IEEE transactions on biomedical circuits and systems*, vol. 9, no. 4, pp. 453–474, 2015.

[7] R. Sarpeshkar, "Analog synthetic biology," *Philosophical Transactions of the Royal Society, A372:20130110*, vol. 2014, DOI: http://dx.doi.org/10.1098/rsta.2013.0110.

[8] ——, *Ultra Low Power Bioelectronics: Fundamentals, Biomedical Applications, and Bio-Inspired Systems*. Cambridge, UK: Cambridge University Press, 2010.

[9] S. Koziol, S. Brink, and J. Hasler, "A neuromorphic approach to path planning using a reconfigurable neuron array IC," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems,*, vol. 22, no. 12, p. 2724–2737, 2014.

[10] S. S. Woo, J. Kim, and ". R. Sarpeshkar, "Digitally programmable cytomorphic chip for simulation of arbitrary biochemical reaction networks", ieee transactions on biomedical circuits and systems," *Vol.*, vol. 12, no. 2, pp. 360–378, Apr. 2018.

[11] J. Kim, S. S. Woo, and R. Sarpeshkar, "Fast and precise emulation of stochastic biochemical reaction networks with amplified thermal noise in silicon chips," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 12, no. 2, pp. 379–389, Apr. 2018.

[12] C. Mead, "Neuromorphic electronic systems," *Proceedings of IEEE*, no. 78, pp. 1629–1636, 1990. [Online]. Available: https://web.stanford.edu/group/brainsinsilicon/documents/MeadNeuroMorphElectro.pdf.

[13] R. Chawla, A. Bandyopadhyay, V. Srinivasan, and P. Hasler, "A 531 nW/MHz 128 $\times$ 32 current-mode programmable analog vector-matrix multiplier with over two decades of linearity," in *IEEE Custom Integrated Circuits Conference*, Oct. 2004, pp. 651–654.

[14] C. Schlottmann and J. Hasler, "A highly dense, low power, programmable analog vector-matrix multiplier: The FPAA implementation," *IEEE Journal of Emerging CAS*, vol. 1, pp. 403–411, 2012.

[15] J. Hasler and H. B. Marr, "Finding a roadmap to achieve large neuromorphic hardware systems," *Frontiers in Neuromorphic Engineering*, pp. 1–29, Sep. 2013.

[16] K. Sanni and A. G. Andreou, "A historical perspective on hardware AI inference, charge-based computational circuits and an 8bit charge-based multiply-add core in 16nm FinFET CMOS," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 9, no. 3532, pp. 532–543, Sep. 2019.

[17] J. Roychowdhury, "Boolean computation using self-sustaining nonlinear oscillators," *in Proceedings of the IEEE*, vol. 103, no. 11, pp. 1958–1969, Nov. 2015. DOI: 10.1109/JPROC.2015.2483061.

[18] T. Wang and J. Roychowdhury, "OIM: Oscillator-based Ising machines for solving combinatorial optimisation problems," in *International Conference on Unconventional Computation and Natural Computation*, Cham, 2019, pp. 232–256.

[19] Y. Yamamoto, "Optical neural network operating at the quantum limit - coherent Ising/XY/recurrent neural network machines," *Photonics in Switching and Computing (PSC)*, vol. 2018, pp. 1–4, 2018. DOI: 10.1109/PS.2018.8751251.

[20] R. L. Wigington, "A new concept in computing," in *Proc. Inst Radio Eng*, Apr. 1959, pp. 516–523.

[21] S. Oshima, "Introduction to Parametron," *Denshi Kogyo*, vol. 4, no. 11, p. 4, Dec. 1955.

[22] D. A. Patterson, T. Anderson, N. Cardwell, R. Fromm, K. Keeton, C. Kozyrakis, R. Thomas, and K. Yelick, "A case for intelligent RAM," *IEEE Micro*, vol. 17, no. 2, pp. 34–44, 1997.

[23] A. S. Cassidy and A. G. Andreou, "Beyond Amdahl's law: An objective function that links multiprocessor performance gains to delay and energy," *IEEE Trans Comput*, vol. 61, no. 8, pp. 1110–1126, Aug. 2012.

[24] M. Gokhale, B. Holmes, and K. Iobst, "Processing in memory: The Terasys massively parallel PIM array," *IEEE Computer*, vol. 28, no. 4, Apr. 1995.

[25] T. L. Sterling and H. P. Zima, "Gilgamesh: A multithreaded processor-in-memory architecture for petaflops computing," in *presented at the Proceedings of the 2002 ACM/IEEE conference on Supercomputing (SC'02)*, ACM/IEEE, 2002.

[26] P. O. Pouliquen, A. G. Andreou, and K. Strohben, "Winner-takes-all associative memory: A Hamming distance vector quantizer," *Analog Integrated Circuits and Signal Processing*, vol. 13, p. 1, May 1997.

[27] R. Karakiewicz, R. Genov, and G. Cauwenberghs, "1.1 TMACS/mW fine-grained stochastic resonant charge-recycling array processor," *IEEE Sensors Journal*, vol. 12, no. 4, pp. 785–792, 2012.

[28] H. Valavi, P. J. Ramadge, E. Nestler, and N. Verma, "A 64-tile 2.4-Mb in-memory-computing CNN accelerator employing charge-domain compute," *IEEE Journal of Solid-State Circuits*, vol. 54, no. 6, pp. 1789–1799, Jun. 2019.

[29] Z. Bian, F. Chudak, R. Israel, B. Lackey, W. G. Macready, and A. Roy, "Discrete optimization using quantum annealing on sparse Ising models," *Frontiers in Physics*, vol. 2, p. 56, 2014.

[30] X. Yin, B. Sedighi, M. Varga, M. Ercsey-Ravasz, Z. Toroczkai, and X. S. Hu, "Efficient analog circuits for boolean satisfiability," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 26, no. 1, pp. 155–167, Jan. 2018. DOI: 10.1109/TVLSI.2017.2754192.

[31] B. Molnár, F. Molnár, M. Varga, Z. Toroczkai, and M. Ercsey-Ravasz, "A continuous-time Max-SAT solver with high analog performance," *Nature Comm*, vol. 9, no. 4864, 2018.

[32] K. Y. Camsari, R. Faria, B. M. Sutton, and S. Datta, "Stochastic p-bits for invertible logic," *Physical Review X*, vol. 20, no. 7, p. 031 014, Jul. 2017. DOI: 10.1103/PhysRevX.7.031014.

[33] B. Sutton, K. Y. Camsari, B. Behin-Aein, and S. Datta, "Intrinsic optimization using stochastic nanomagnets," *Scientific reports*, vol. 15, no. 7, p. 1, Mar. 2017.

[34] K. Y. Camsari, P. Debashis, V. Ostwal, A. Z. Pervaiz, T. Shen, Z. Chen, S. Datta, and J. Appenzeller, "From charge to spin and spin to charge: Stochastic magnets for probabilistic switching," *Proceedings of the IEEE*, vol. 12, no. 108, p. 8, Feb. 2020.

[35] B. Sutton, R. Faria, L. A. Ghantasala, R. Jaiswal, K. Y. Camsari, and S. Datta, "Autonomous probabilistic coprocessing with petaflips per second," *IEEE Access*, vol. 8, pp. 57 238–15 725, 2020.

[36] A. Zadehgol and A. C. Cangellaris, "Isotropic spatial filters for suppression of spurious noise waves in sub-gridded FDTD simulation," *IEEE Transactions Antenna Propagation*, vol. 59, no. 9, pp. 3272–3279, 2011.

[37] A. Zadehgol, "Guest editorial for the special issue on signal and power integrity of microelectronic networks through modeling and simulation of fields and devices," *International Journal of Numerical Modelling: Electronic Networks, Devices and Fields*, p. 2457, 2018.

[38] J. S. Toll, "Causality and the dispersion relation: Logical foundations," *American Physical Society*, vol. 104, no. 6, pp. 1760–1770, 1956. DOI: 10.1103/PhysRev.104.1760.

[39] R. Choupanzadeh and A. Zadehgol, "Stability, causality, and passivity analysis of canonical equivalent circuits of improper rational transfer functions with real poles and residues," *IEEE Access*, vol. 8, pp. 125 149–125 162, 2020. DOI: 10.1109/ACCESS.2020.3007854.

[40] A. Zadehgol, "A semi-analytic and cellular approach to rational system characterization through equivalent circuits," *International Journal of Numerical Modelling: Electronic Networks, Devices and Fields*, vol. 29, no. 4, pp. 637–652, 2016.

[41] B. Brogliato, R. Lozano, B. Maschke, and O. Egeland, *Dissipative Systems Analysis and Control, Theory and Applications, 2nd ed*. p. 22: Springer, 2007.

[42] A. Zadehgol, "Passivity considerations for sub-gridded FDTD with discrete complex wave impedance," in *EMC Europe*, 2016, pp. 72–74.

[43] R. C. Dorf and R. H. Bishop, *Modern control systems*. Harlow: Pearson Education Limited, 2017, 371-373.

[44] A. Zadehgol, "A frequency-independent and parallel algorithm for computing the zeros of strictly proper rational transfer functions," *Appl Math Comput*, vol. 274, pp. 229–236, 2016.

[45] N. Yamin and A. Zadehgol, "Verification and enforcement of passivity through direct minimal modification of equivalent circuits," in *EMC Eur*, 2016, pp. 756–759.

[46] A. Zadehgol, "An impedance transfer function formulation for reduced-order macromodels of subgridded regions in FDTD," *IEEE Transactions Antennas Propagation*, vol. 65, no. 1, pp. 401–404, Jan. 2017.

[47] ——, "Deterministic reduced-order macromodels for computing the broadband radiation-field pattern of antenna arrays in FDTD," *IEEE Transactions Antenna Propagation*, vol. 64, no. 6, pp. 2418–2430, Jun. 2016.

[48] ——, "Stochastic reduced-order electromagnetic macromodels in FDTD," *IEEE Transactions Antenna Propagation*, vol. 64, no. 8, pp. 3496–3508, Aug. 2016.

[49] A. Zadehgol, H. Lei, and B. K. Johnson, "A methodology for remote sensing inter-turn fault events in power system air-core reactors, via simulation of magneto quasi-static fields in 2D FDTD," *IEEE Access*, pp. 1–1, 2020. DOI: 10.1109/access.2020.3024927.

[50] A. Zadehgol, "An efficient approximation for arbitrary port suppression of multiport scattering parameters," *International Journal of Numerical Modelling-Electronic Networks Devices and Fields*, vol. 27, no. 1, pp. 164–172, Jan. 2014.

[51] A. Zadehgol, A. C. Cangellaris, and P. L. Chapman, "A model for the quantitative electromagnetic analysis of an infinitely long solenoid with a laminated core," *International Journal of Numerical Modelling-Electronic Networks Devices and Fields*, vol. 24, no. 3, pp. 244–256, 2011.

[52] H. Jaeger, "Exploring the landscapes of "computing": Digital, neuromorphic, unconventional–and beyond," preprint, 2020.

[53] J. J. Y. Teo and R. Sarpeshkar, "The merging of biologic and electronic circuits," *iScience*, vol. 23, no. 11, 2020.

[54] T. Hylton, "Thermodynamic neural network," *Entropy*, vol. 22, no. 3, p. 256, 2020.

[55] B. Scellier, "A deep learning theory for neural networks grounded in physics," arXiv, preprint, 2021. arXiv: 2103.09985.

[56] J. Hasler, S. Kim, and A. Natarajan, "Enabling energy-efficient physical computing through analog abstraction and IP reuse," *Journal of Low Power Electronics Applications*, pp. 1–23, Dec. 2018.

[57] S. S. Woo and R. Sarpeshkar, "A cytomorphic chip for quantitative modeling of fundamental bio-molecular circuits," *IEEE Transactions in Biomedical Circuits and Systems, Special Issue on Synthetic Biology*, vol. 9, no. 4, pp. 527–542, 2015.

[58] R. Daniel, J. R. Rubens, R. Sarpeshkar, and T. K. Lu, "Synthetic analog computation in living cells," *NATURE*, vol. 497, no. 7451, pp. 619–623, 2013. DOI: 10.1038/nature12148.

[59] R. Sarpeshkar, C. Salthouse, J. J. Sit, M. Baker, S. Zhak, T. Lu, L. Turicchia, and S. Balster, "An ultra-low-power programmable analog bionic ear processor," *IEEE Transactions on Biomedical Engineering*, vol. 52, no. 4, pp. 711–727, Apr. 2005.

[60] R. Hahnloser, R. Sarpeshkar, M. Mahowald, R. Douglas, and S. Seung, "Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit," *NATURE*, vol. 405, no. 22, Cover article, Jun. 2000.

[61] Y. Yamamoto, T. Leleu, S. Ganguli, and H. Mabuchi, "Coherent Ising machines—-quantum optics and neural network perspectives," *Appl Phys. Let*, vol. 117, no. 16050, p. 1, 2020.

[62] A. Papp, W. Porod, and G. Csaba, "Perspectives of using spin waves for computing and signal processing," *Physics Letters*, vol. 381, no. 17, pp. 1471–1476, May 2017. DOI: 10.1016/j.physleta.2017.02.042. [Online]. Available: https://doi.org/10.1016/j.physleta.2017.02.042.

[63] G. Csaba and W. Porod, "Coupled oscillators for computing: A review and perspective," *Applied Physics Reviews*, vol. 7, p. 011 302, 2020. [Online]. Available: https://doi.org/10.1063/1.5120412.

[64] G. Csaba, A. Papp, and W. Porod, "Spin-wave based realization of optical computing primitives," *J Appl. Phys*, vol. 115, no. 17, p. 17, 2014.

[65] Y. Khivintsev, M. Ranjbar, D. Gutierrez, H. Chiang, A. Kozhevnikov, Y. Filimonov, and A. Khitun, "Prime factorization using magnonic holographic devices," *Journal of Applied Physics*, vol. 120, Sep. 2016. DOI: 10.1063/1.4962740.

[66] M. Balynsky, A. Kozhevnikov, Y. Khivintsev, T. Bhowmick, D. Gutierrez, H. Chiang, G. Dudko, Y. Filimonov, G. X. Liu, C. L. Jiang, A. A. Balandin, R. Lake, and A. Khitun, "Magnonic interferometric switch for multi-valued logic circuits," *Journal of Applied Physics*, vol. 121, Jan. 2017. DOI: 10.1063/1.4973115.

[67] A. Raychowdhury, A. Parihar, G. H. Smith, V. Narayanan, G. Csaba, M. Jerry, W. Porod, and S. Datta, "Computing with networks of oscillatory dynamical systems," *Proceedings of the IEEE, Year:*, vol. 107, no. 1, pp. 73–89, 2019. DOI: 10.1109/JPROC.2018.2878854.

[68] A. Mallick, M. K. Bashar, D. S. Truesdell, B. H. Calhoun, S. Joshi, and N. Shukla, "Using synchronized oscillators to compute the maximum independent set," *Nature communications*, vol. 11, no. 1, pp. 1–7, 2020.

[69] M. R. Pufall, W. H. Rippard, G. Csaba, D. E. Nikonov, G. I. Bourianoff, and W. Porod, "Physical implementation of coherently coupled oscillator networks," *IEEE J. Exploratory Solid-State Computational Devices and Circuits*, vol. 1, pp. 76–84, 2015.

[70] D. E. Nikonov, G. Csaba, W. Porod, T. Shibata, D. Voils, D. Hammerstrom, I. A. Young, and G. I. Bourianoff, "Coupled-oscillator associative memory array operation for pattern recognition," *IEEE J Exploratory Solid-State Computational Devices and Circuits*, vol. 1, pp. 85–93, 2015.

[71] J. Hasler, "Large-scale field programmable analog arrays," *Proceedings of IEEE*, vol. 108, no. 8, pp. 1283–1302, Aug. 2020. [Online]. Available: http://hasler.ece.gatech.edu/FPAA_IEEEXPlore_2020.pdf.

[72] J. Hasler, A. Natarajan, S. Shah, and S. Kim, "Circuit implementations teaching a junior level circuits course utilizing the SoC FPAA," *IEEE Microelectronics Systems Education*, pp. 7–10, May 2017.

[73] T. Wang and J. Roychowdhury, "Design tools for oscillator-based computing systems," in *52nd Design Automation Conference (DAC)*, San Francisco CA, USA: ACM/EDAC/IEEE, 2015, pp. 1–6.

[74] C. Mead and L. Conway, *Introduction to VLSI systems*. Addison-Wesley, 1980.

Chapter 7

Application Domains beyond Circuits and Electronic Systems

7.1 Background

EDA is a field that brings scientific theory (such as algorithm design and theorem proving) to the design of electronic circuits and systems, and thus it is a field of *design science*. The fundamental concepts of EDA include formalization, refinement, design reuse and simulation. EDA methodologies and tools have achieved great success in managing the enormous complexity in electronic systems by building on the general principles of abstraction (bottom-up) and refinement (top-down) as well as decomposition and composition for both verification and design. Though new advances in EDA are required to overcome the challenges brought by future electronic systems (see Chapter 3), the design science principles developed in the EDA field can also serve other engineered systems.

There are many (existing and new) applications that can greatly benefit from the EDA technology base and tool set. Examples include AI on the edge and in the cloud, cyber-physical and IoT systems, secure multi-party computation and homomorphic encryption, sustainable computing with full lifecycle assessment goals and metrics, drug discovery and preventative healthcare, etc. Furthermore, with the looming end of conventional scaling, many beyond-CMOS technologies and computing paradigms are being aggressively pursued, including in- and near-memory computing, stochastic computing, superconductive computing, hyperdimensional computing, quantum computing, etc. Finally, new objectives and technology-driven/application-specific constraints are transforming the standard formulations of typical EDA problems, sometimes fundamentally changing the nature of the underlying problems. Examples include preservation of privacy and establishing trust in computing systems, accounting for gate-level pipelining and full path balancing requirements of some emerging computation fabrics, etc. There is a need for developing full-stack integrated solutions to these new problems.

When considering new application domains, it is important to answer the following questions.

- What new applications will benefit most from circuit/system design and design automation solutions, and from which subset of conventional EDA tools and in what ways?
- What new applications will set new functionality and/or new performance and scalability requirements for the circuit/system design and design automation solutions, and in what ways?
- What are the unique challenges brought about by some new application domains, which are not adequately addressed in conventional EDA methodologies?
- As we move up to the design stack all the way to applications, is it still possible to have tools that can be applied to a variety of application domains? If yes, how would one categorize the different application domains based on their needs for design tools? If not, should such tool development effort belong to the end application domain or do design and design automation tools still play a key enabling technology role?

7.2 Adjacent application domains

Technology extensions. Design and design automation will continue to play an indispensable role in the development of ICs based not only on advanced CMOS technology nodes (5 nm and below) but also many flourishing beyond CMOS emerging technologies ranging from phase change to resistive switching arrays, from spintronic to ferroelectric devices, and from superconductive Josephson junctions to nanophotonic devices (e.g., [1]–[5], and also see Chapter 4). In addition, next generation EDA tools must support heterogeneous integration of these disparate technologies across many different specialized platforms in support of large scale multi-physics based heterogeneous systems. It is well accepted that the many local and global sources of variability in extreme-scaled CMOS devices and circuits as well as the inherently random nature of many beyond-CMOS technologies demand that EDA platforms and tools properly model and cope with deterministic and stochastic behaviors and noisy inputs/outputs while supporting approximate (imprecise) computations. EDA tools will need to provide cohesive design flows combining a variety of emerging technologies across multiple platforms (such as 3-D integration, silicon in a package (SiP), and wafer-scale integration (WSI).

A host of novel systems are possible. These systems will be heterogeneous in nature and will utilize a variety of emerging, exotic technologies while coexisting with deeply scaled CMOS. Integrated design capabilities will need to be developed that can support this sort of heterogeneous system across multiple abstraction layers and technologies. Synchronization will need to naturally include a variety of timing schemes, ranging from fully synchronous to self-timed systems, and the relevant EDA tools will need to naturally support these heterogeneous timing paradigms. Power delivery and relevant converters will need to be integrated on-chip to provide excessive current levels while managing power ripple and noise.

Example applications could include a combination of CMOS, single flux quantum, and qubits where signals are passed from room temperature through nitrogen and helium to millikelvin temperatures. Superconducting circuits are expected to become a significant part of the energy and performance solution for the stationary server farm community, particularly targeting cloud computing [6]. Mobile applications will only become more pervasive with frequencies approaching a terahertz. Mixed-signal and RF circuits will be integrated with on-chip antennas to form powerful compact communications devices. All of these applications will require design algorithms, techniques, and tools that support the full spectrum of capabilities, synthesis, simulation, modeling, verification, and test.

Systems of systems. Computational sciences and large-scale system simulation and validation remain as key challenges for EDA tools. The emergence of large-scale systems—systems-of-systems, large-scale distributed information sharing and computing platforms—introduces new challenges for EDA. Both EDA industry and academic research have traditionally focused on integrated circuits, and more particularly the hardware components of those integrated circuits. A recent article highlights the widespread view that computer system design has expanded its focus from hardware to systems [7]:

> ...the industry continues to move away from the idea that lithography will provide additional performance improvements, and towards a model that prizes a multi-disciplinary approach to semiconductor performance improvement. Tightening the linkages between hardware and software and squeezing out inefficiencies is how companies are pushing performance forward these days.

Addressing systems-of-systems design requires expanding the scope of EDA in the following aspects:

- Modern systems-of-systems integrate components from multiple hardware technologies: multiple fabrication nodes, mixed-signal, large amounts of memory. EDA must span component boundaries and technology boundaries.
- Systems-of-systems that connect to physical plants must be designed within the context of those plants. Design verification methods that span the space of simulation and formal methods must be able to take into account

complex computing platforms, software, and physical plants.

- Systems-of-systems rely on large amounts of software to provide real-time, low-power, reliable functions. Software and hardware must be designed together.
- Systems-of-systems must optimize themselves on-the-fly at run-time; design time optimization is no longer sufficient due to system complexity, complex workloads, and aging effects.
- Safety and security are key design goals that must be baked in, not bolted on [8].

From cloud to edge. With the shift from cloud to edge devices, connectivity, distributed intelligence, and collaborative computation are becoming paramount, and EDA must step up its efforts to offer design platforms that enable such interactivity and sharing without sacrificing the user data or intellectual property. This is also one reason that there must be a push toward developing support for privacy preservation and enhancing technologies both at design and run time.

Machine learning (ML) is becoming a dominating workload in terms of shifting from cloud to edge. ML training and inference can incur huge computational, communication, memory, and energy costs, where appropriate designs can play a substantial role in reducing these demands. For instance, reduction in computational requirements can be achieved through reduced precision, while reducing latency can be achieved through structured sparsity [9]. An interesting problem in this context is accurate prediction of how models developed using various optimization techniques (to be employed during training) will perform when used at inference time on specific hardware (which was not used for training). Moreover, an important question is whether hardware can expose useful information (to higher layers) to facilitate such predictions. To this end, there is a need for effective model search techniques that incorporate computational complexity and hardware characteristics.

It is clear that success of machine learning models is largely driven by data; however, this is potentially detrimental to preserving privacy, a critical need for achieving societal acceptance of many applications. Of importance here is being able to quantify the trade-off between privacy of users' data and the corresponding utility of information obtained, while also providing guarantees for the level of privacy achieved [10]. The shift from cloud to edge further exacerbates these challenges (as we can no longer afford to transfer all data to the cloud), leading to the need for even more efficient models, ability to deal with (potentially) substantial noise in data collection at the edge, leading to the need for hardware support for data processing at the edge, while also optimizing the entire processing pipeline, from the sensors collecting data, to the edge, to the cloud [11], [12].

Autonomous systems. Other than in case of systems automation, an autonomous system has flexibility in decision making, in order to reach goals rather than following an algorithm or a state machine. To enable such self-governance, autonomous systems establish knowledge and an image of itself and its environment, called self-awareness [13]. From a scientific perspective, autonomous systems are at the intersection of automation, cyber-physical system (CPS), and artificial intelligence [14].

Autonomous systems are an emerging topic in many application areas, in transport and mobility (automated vehicles, UAV, space robotics) all the way to large-scale CPS, such as in smart grids or smart buildings. In these applications, autonomous systems functions are safety critical and/or require high availability. Also, any form of machine learning entails behavioral changes with unexplored side effects on other parts of a system that are even harder to control under system autonomy than in traditional automation. What is therefore needed, is research into assured autonomy (e.g., DARPA's BAA (HR001117S0045): Assured Autonomy). Assured autonomy must be approached as an engineering process that is compatible with current high assurance design, such as defined in safety standards. Bringing high assurance design to systems autonomy is a huge challenge for both the hardware/software architectures and the related autonomous systems design processes [15]. One approach to address this challenge is to establish a layer of autonomy supervision that uses well- understood mechanisms to bound the behavior of the autonomous

system, as a further development of the established safety layers.

While traditional design separates a lab design phase from a field operation phase, assured autonomy will require monitored systems evolution in the field. To maintain design quality and to safeguard evolution in the field operation phase, EDA support should be extended to the field with highly automated versions of EDA tool functionality including the related model base for self- and context-modeling. New tools for dependency and automated failure analysis will be needed, as well as for model adaptation and model error detection. The lab design phase will be equally affected. Behavioral goals and constraints guiding the autonomous system must be formulated and verified to prepare for the in-field phase. We need tools for synthesis and configuration of autonomy supervision components and their in-field operation. Without such EDA tools, the permitted behavioral dynamics of critical autonomous systems will be severely limited. In other terms, EDA will be key to make assured autonomy happen.

New design concerns. With respect to new design concerns such as privacy, resiliency, sustainability of the computing fabrics, it was argued that these concerns are on par with more traditional design concerns such as area, speed, reliability, and power efficiency, and that EDA platforms and tools must model and enable meaningful tradeoffs among these often-conflicting concerns in a way that would empower not only the product developers but also the end users of the electronic products. A key to achieving this visibility and effective control over a multi-dimensional, dynamically evolving design optimization space is the creation of better links and hooks from higher levels of design abstraction to low level design details that set the true hardware performance.

As an example, consider the problem of neural architecture search (NAS), which aims to maximize neural network performance while minimizing compute resources (multiply/add, number of weights, etc.), see e.g., [16]. The issue here is that the resource count is not necessarily representative of the system performance after hardware mapping. For instance, a neural network with a small number of multiply/adds may consume a large amount of energy if its topology does not lead to efficient data re-use. Studying this data flow and compute scheduling problem at the hardware level is by itself a complex task [17] and coupling it with a neural architecture search is a significant challenge that must be addressed in the future. The problem becomes even more complex when the non-idealities of analog hardware fabrics and their trade-offs must be considered (as for instance, in in-memory computing) [18].

An overarching problem in modern system design is that high-performance hardware is becoming increasingly domain specific [19]. This leads to issues in amortizing design complexity and justifying a full-stack optimization (as discussed above) for a specific application with limited sales volume. Going forward, the community must identify common denominators between the various applications and application domains to develop platforms, tools and optimization methods that are broadly applicable. At the front of hardware platforms, this is already taking place with the exploration of CGRAs (Coarse Grain Reconfigurable Architectures) [20]. However, the design tools and optimization methods for efficient mapping of arbitrary applications onto CGRAs are still at their infancy.

7.3 Beyond adjacent applications domains

There are a number of application domains that can benefit tremendously from the systematic and methodical design automation process that has been developed in the last five decades for electronic systems. In essence, any engineered system (e.g., smart buildings [21] and electric vehicles [22]) can apply the design science principles developed in the EDA field. The confluence of design automation with these application domains bring new challenges and opportunities. Below, we discuss some representative application domains that are “farther-away” from electronic systems.

Network design automation. Traditionally the field of network design and analysis and electronic design and analysis rarely interacts. However, as large-scale networks become prevalent and network sizes continue to grow, the

complexity of network configurations also increases rapidly. Network reliability, security, load distribution and cost all equally important metrics in network design. These trends have led new network analysis and verification techniques which are inspired by EDA methodologies [23].

For networks with hundreds of thousands of nodes, network outages are quite common. Verification technologies supporting proactive prevention of potential network disruption are highly desirable. However, there can be millions of routing rules in such large networks, which makes it difficult to verify. Motivated by finite state machine verification, a formal geometric model of network forwarding called Header Space was introduced, where routers are considered as states. The header space analysis can leverage the concepts in finite state machine verification as well as systematic reduction of network size by leveraging regular structures inherent in the network, much like the modular design approach in EDA. Network specific structures are integrated to the analysis in order to scale better than off-the-shelf model checkers [24]. One specific reduction technique is based on symmetry in the rule sets of symmetrically placed routers, similar to exploiting logical Kripke structure in [25].

Similar to electronic system design, formal specification of networks may not be always available, and available networks specifications tend to be incomplete and ambiguous. In a very recent work [26], network data mining is proposed to find bugs (i.e., misconfigurations) without formal specification of intended behavior of the network. Through the introduction of a general approach to outlier detection, an automatic template inference is achieved, where the templates model the intentional differences as variations within a template and erroneous differences as variations across templates [26]. Such techniques could also benefit electronic system design.

The term, “network design automation”, has been adopted by some network researchers (e.g., it is the title of a large project funded by NSF, CNS program in 2019). It is clear that there is quite some synergy between NDA and EDA. In a plenary talk by George Varghese (an expert in the network research area) at this workshop, he has outlined a wish list of EDA design tools which include automatic test packets, debuggers (how to “step” through network), timing verification for real time traffic, a formal description language like Verilog for network configurations, and scalable specifications that can cover many different network types. clearly, close research collaboration among EDA and NDA researchers and jointly sponsored research projects would benefit both application domains, where the EDA theory and methodologies would form the common foundational fabric.

Biology and lab-on-chip. EDA methodologies have demonstrated initial success in the synthetic biology and lab-on-chip fields [27]–[32]. It is important that EDA continues to develop design flows and tools that support such critical applications.

Synthetic biology was built upon genetic engineering by adding the engineering principles of standards, abstraction, and decoupling [33]. These principles are the cornerstones of EDA, so it should not be surprising that EDA research and EDA researchers have been responsible for the new field of genetic design automation (GDA) that has produced the necessary software tools to support this domain. The EDA community has participated in the development of standards, such as the Synthetic Biology Open Language (SBOL) [34] that can be used to share data between research groups and software tools via repositories such as SynBioHub [35]. Modeling, analysis and analysis tools have been developed (e.g., [36], [37]. As synthetic biology expands its commercial application, EDA researchers will be in ever more demand to continue to support high-throughput design processes.

EDA has contributed to bridging the gap between advances in microfluidics lab-on-chip technology and its adoption for microbiology. EDA research has been licensed by biotech companies such as Illumina, Baebies, and GenMark, with applications to immunoassays, sample preparation, and health screening for newborns. Today’s design automation solutions for lab-on-chip address not only the classical problems of synthesis, routing, reliability, and design-for-test, but they also incorporate domain-specific constraints such as limited stock solutions, uncertainties in bioassay

sample pathways, and type-driven single-cell analysis, and dynamic adaptation based on machine learning [31], [38]. The next generation of EDA solutions for microfluidic labs-on-chip must enable an experimental framework for quantitative-analysis studies using a lab-on-chip that utilizes the smallest amount of samples to obtain answers to questions in biology with the highest possible precision.

From a biology perspective, EDA concepts can be leveraged to develop an experimental framework for: (i) identification of biomolecules—also known as up-stream analysis—that exhibit specific or abnormal biological behavior (e.g., fluorescence-based gene expression/suppression due to enzymatic reaction); (ii) causative exploration of biomolecules—also known as down-stream analysis—that contribute to the in vivo interactions leading to such biological behavior (e.g., the impact of protein binding to DNA at specific genomic regions). Such causative exploration requires a method for indexing/barcoding samples at the end of up-stream analysis to account for cellular heterogeneity. As a result, there is a need to advance EDA for lab-on-chip to support specification-driven experiment design, sample labeling and differentiation, and multi-sample experimentation. The first stage, i.e., experiment design, is carried out before experimentation and it seeks co-optimization of experiment goals (e.g., amount of input sample, on-chip resources, and sequencing coverage). A decision at this stage impacts the protocol procedure. The second stage, i.e., sample labeling and differentiation, is focused on developing scalable methods for differentiation of heterogeneous samples within down-stream analysis. The third stage, i.e., multi-sample experimentation, represents the implementation of protocol procedures using multiple sample pathways in order to obtain quantitative results; this stage applies to both up-stream and down-stream analyses.

Formal models are argued to be an excellent way to store and share knowledge on biological systems, and to reason about such systems [39]. Researchers have called for a good computational/operational model to explain the mechanisms behind a biological system [40], [41]. Computational/operational models (such as Petri nets) are executable and may mimic biological processes better than a mathematical/denotational model with a set of equations [40]. However, an executable model must incorporate all the complexity of the biological system to make it an effective predictive model. The state of our knowledge about any biological system is so poor that it would be hard to come up with an executable model. It is easier to capture properties of a biological system than the actual state transition model of the underlying system. Even if the structure and mechanism of the underlying biological system is unclear, one can still think about simple properties using state variables that can be measured extensively using modern high-throughput technologies.

Biomedical systems and drug design. Biomedical systems, drug design, and Healthcare are important technology drivers and important new applications for EDA as well. There are many similarities between biomedical system analysis and electronic system analysis. Finding a bug in an electronic system involve simulation and verification of input/output relationships. Studying biomedical systems also involve providing some input to the system, change the underlying system one variable at a time and observe the output. This is exactly how EDA technologies are used to study electronic system. Therefore, EDA technologies such as finding a bug using a series of inputs can be useful to study biomedical system. This can help us identify important molecular signals that drives human diseases.

Following are examples of the application of formal methods in biomedical systems to study differentiation. Differentiation is a concept from developmental biology where a cell changes from one type to another. Fundamentally, it is very similar to a state transition system. Differentiation has been modeled mathematically using a variety of approaches such as Boolean Network Extension (BNE) [42], Boolean Implication Network [43]–[45], and Boolean simulation framework [46]. A formal CAD approach has led to discovery of biomarkers of colon epithelial differentiation across gene-expression arrays from patients with stage II/III colorectal cancers (CRCs); that identified a subgroup of patients with high-risk stage II colon cancer who appeared to benefit from adjuvant chemotherapy [47]. This work is an example of an impactful study that uses the concept from EDA to transform understanding of the biomedical system.

A potential new application domain where EDA-like techniques can be applied is discovery and development of new drug molecules. Drug discovery has hit a major productivity crisis. Getting a single drug to market takes an arduous 10 to 12 years, with an estimated price tag of nearly $3 billion [48]. This slowing pace and rising cost of R&D has recently been coined Eroom's law, so named because it's the opposite to Moore's law, whereby computing power doubles and cost is halved roughly every 18 months.

To address the productivity crisis, the drug design paradigm is being re-thought by using a combination of computation and automated chemistry platforms. Computational techniques such as deep learning have been used to identify new drug-like molecules [49]. In addition, synthetic chemistry platforms are now enabling the fully automated multi-step synthesis of quite complex molecules at scales from nanograms to grams, and at unprecedented speeds [50]. However, machine learning-based computational methods are still in their infancy and do not provide a good framework to capture existing expert knowledge and consequently require large amounts of data which limits their utility. The entire chemical space of molecules with favorable pharmacokinetic properties in terms of absorption and distribution has been estimated to be 10^{60} molecules and is far too large for exhaustive enumeration. EDA experts can play a key role in this application domain as there are numerous parallels between the drug discovery process and IC chip design and manufacturing processes in that chemical manufacturing rules and expert knowledge can be abstracted into "design" rules that in combination with efficient heuristics can be used to make the search more tractable. EDA's deep expertise in creating domain-specific abstractions, algorithm development and optimization would facilitate the development of a multi-stage computational software pipeline that can greatly expedite new drug discovery.

General engineered systems. The methodology and approaches of EDA can benefit other complex, engineered systems. "Engineering system design automation (ESDA)", which can capture the application domains with identification of proper path to manufacturing, are the natural evolution of the current EDA solutions. ESDA must deal with the huge proliferation of complex, large, cyber-physical systems where software is as important as hardware in determining the functionality and performance of the target engineered system and where regulatory requirements new design concerns (such as privacy and sustainability) must be met. EDAS must not only help with translating and optimizing systems from the specification stage to manufactured product stage but also be able to help accommodate revisions after product release.

Bibliography

[1] C. Batten, A. Joshi, V. Stojanovć, and K. Asanović, "Designing chip-level nanophotonic interconnection networks," in *Integrated Optical Interconnect Architectures for Embedded Systems*, Springer, 2013, pp. 81–135.

[2] M. Kazemi, E. Ipek, and E. G. Friedman, "Adaptive compact magnetic tunnel junction model," *IEEE Transactions on Electron Devices*, vol. 61, no. 11, pp. 3883–3891, 2014.

[3] L. Amarú, P.-E. Gaillardon, S. Mitra, and G. De Micheli, "New logic synthesis as nanotechnology enabler," *Proceedings of the IEEE*, vol. 103, no. 11, pp. 2168–2195, 2015.

[4] D. Ielmini and H.-S. P. Wong, "In-memory computing with resistive switching devices," *Nature Electronics*, vol. 1, no. 6, pp. 333–343, 2018.

[5] A. Aziz, E. T. Breyer, A. Chen, X. Chen, S. Datta, S. K. Gupta, M. Hoffmann, X. S. Hu, A. Ionescu, M. Jerry, *et al.*, "Computing with ferroelectric FETs: Devices, models, systems, and applications," in *2018 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, IEEE, 2018, pp. 1289–1298.

[6] G. Krylov and E. G. Friedman, "Design methodology for distributed large-scale ERSFQ bias networks," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 28, no. 11, pp. 2438–2447, 2020.

[7] J. Hruska, "AMD will support smart access memory on ryzen 3000 CPUs for gaming," *Extreme Tech*, Mar. 2021.

[8] M. Wolf and D. Serpanos, "Safety and security in cyber-physical systems and internet-of-things systems," *Proceedings of the IEEE*, vol. 106, no. 1, pp. 9–20, 2017.

[9] S. Dey, Y. Shao, K. M. Chugg, and P. A. Beerel, "Accelerating training of deep neural networks via sparse edge processing," in *International Conference on Artificial Neural Networks*, Springer, 2017, pp. 273–280.

[10] A. Marchisio, M. A. Hanif, F. Khalid, G. Plastiras, C. Kyrkou, T. Theocharides, and M. Shafique, "Deep learning for edge computing: Current trends, cross-layer optimizations, and open research challenges," in *2019 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, IEEE, 2019, pp. 553–559.

[11] C.-C. Hung, G. Ananthanarayanan, P. Bodik, L. Golubchik, M. Yu, P. Bahl, and M. Philipose, "Videoedge: Processing camera streams using hierarchical clusters," in *2018 IEEE/ACM Symposium on Edge Computing (SEC)*, IEEE, 2018, pp. 115–131.

[12] D. Zhang, Y. Ma, C. Zheng, Y. Zhang, X. S. Hu, and D. Wang, "Cooperative-competitive task allocation in edge computing for delay-sensitive social sensing," in *2018 IEEE/ACM Symposium on Edge Computing (SEC)*, IEEE, 2018, pp. 243–259.

[13] N. Dutt, C. S. Regazzoni, B. Rinner, and X. Yao, "Self-awareness for autonomous systems," *Proceedings of the IEEE*, vol. 108, no. 7, pp. 971–975, 2020.

[14] M. Möstl, J. Schlatow, R. Ernst, N. Dutt, A. Nassar, A. Rahmani, F. J. Kurdahi, T. Wild, A. Sadighi, and A. Herkersdorf, "Platform-centric self-awareness as a key enabler for controlling changes in CPS," *Proceedings of the IEEE*, vol. 106, no. 9, pp. 1543–1567, 2018.

[15] S. Saidi, D. Ziegenbein, J. V. Deshmukh, and R. Ernst, "EDA for autonomous behavior assurance," in *2020 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*, IEEE, 2020, pp. 1–3.

[16] H. Cai, L. Zhu, and S. Han, *Proxylessnas: Direct neural architecture search on target task and hardware*, 2018.

[17] X. Yang, M. Gao, Q. Liu, J. Setter, J. Pu, A. Nayak, S. Bell, K. Cao, H. Ha, P. Raina, *et al.*, "Interstellar: Using Halide's scheduling language to analyze DNN accelerators," in *Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems*, 2020, pp. 369–383.

[18] Y. Ding, W. Jiang, Q. Lou, J. Liu, J. Xiong, X. S. Hu, X. Xu, and Y. Shi, "Hardware design and the competency awareness of a neural network," *Nature Electronics*, vol. 3, no. 9, pp. 514–523, 2020.

[19] W. J. Dally, Y. Turakhia, and S. Han, "Domain-specific hardware accelerators," *Communications of the ACM*, vol. 63, no. 7, pp. 48–57, 2020.

[20] R. Prabhakar, Y. Zhang, and K. Olukotun, "Coarse-grained reconfigurable architectures," in, B. Murmann and B. B. Hoefflinger, Eds., Springer, 2020, pp. 227–246.

[21] R. Jia, B. Jin, M. Jin, Y. Zhou, I. C. Konstantakopoulos, H. Zou, J. Kim, D. Li, W. Gu, R. Arghandeh, *et al.*, "Design automation for smart building systems," *Proceedings of the IEEE*, vol. 106, no. 9, pp. 1680–1699, 2018.

[22] C. Lv, X. Hu, A. Sangiovanni-Vincentelli, Y. Li, C. M. Martinez, and D. Cao, "Driving-style-based codesign optimization of an automated electric vehicle: A cyber-physical system approach," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 4, pp. 2965–2975, 2018.

[23] G. Varghese, "Network verification—when Clarke meets Cerf," in *2016 Formal Methods in Computer-Aided Design (FMCAD)*, IEEE, 2016, pp. 3–3.

[24] G. D. Plotkin, N. Bjørner, N. P. Lopes, A. Rybalchenko, and G. Varghese, "Scaling network verification using symmetry and surgery," *ACM SIGPLAN Notices*, vol. 51, no. 1, pp. 69–83, 2016.

[25] E. A. Emerson and A. P. Sistla, "Symmetry and model checking," *Formal methods in system design*, vol. 9, no. 1-2, pp. 105–131, 1996.

[26] S. K. R. Kakarla, A. Tang, R. Beckett, K. Jayaraman, T. Millstein, Y. Tamir, and G. Varghese, "Finding network misconfigurations by automatic template inference," in *17th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 20)*, 2020, pp. 999–1013.

[27] C. J. Myers, "Computational synthetic biology: Progress and the road ahead," *IEEE Transactions on Multi-Scale Computing Systems*, vol. 1, no. 1, pp. 19–32, 2015.

[28] T. Nguyen, T. S. Jones, P. Fontanarrosa, J. V. Mante, Z. Zundel, D. Densmore, and C. J. Myers, "Design of asynchronous genetic circuits," *Proceedings of the IEEE*, vol. 107, no. 7, pp. 1356–1368, 2019.

[29] K. Hu, F. Yu, T.-Y. Ho, and K. Chakrabarty, "Testing of flow-based microfluidic biochips: Fault modeling, test generation, and experimental demonstration," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 33, no. 10, pp. 1463–1475, 2014.

[30] M. Ibrahim, K. Chakrabarty, and U. Schlichtmann, "Synthesis of a cyberphysical hybrid microfluidic platform for single-cell analysis," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 38, no. 7, pp. 1237–1250, 2018.

[31] M. Ibrahim, A. Sridhar, K. Chakrabarty, and U. Schlichtmann, "Synthesis of reconfigurable flow-based biochips for scalable single-cell screening," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 38, no. 12, pp. 2255–2270, 2018.

[32] Z. Zhong, Z. Li, K. Chakrabarty, T.-Y. Ho, and C.-Y. Lee, "Micro-electrode-dot-array digital microfluidic biochips: Technology, design automation, and test techniques," *IEEE transactions on biomedical circuits and systems*, vol. 13, no. 2, pp. 292–313, 2018.

[33] D. Endy, "Foundations for engineering biology," *Nature*, vol. 438, no. 7067, pp. 449–453, 2005.

[34] J. A. McLaughlin, J. Beal, G. Mısırlı, R. Grünberg, B. A. Bartley, J. Scott-Brown, P. Vaidyanathan, P. Fontanarrosa, E. Oberortner, A. Wipat, *et al.*, "The synthetic biology open language (SBOL) version 3: Simplified data exchange for bioengineering," *Frontiers in Bioengineering and Biotechnology*, vol. 8, 2020.

[35] J. A. McLaughlin, C. J. Myers, Z. Zundel, G. Mısırlı, M. Zhang, I. D. Ofiteru, A. Goni-Moreno, and A. Wipat, "SynBioHub: A standards-enabled design repository for synthetic biology," *ACS synthetic biology*, vol. 7, no. 2, pp. 682–688, 2018.

[36] L. Watanabe, T. Nguyen, M. Zhang, Z. Zundel, Z. Zhang, C. Madsen, N. Roehner, and C. Myers, "iBioSim 3: A tool for model-based genetic circuit design," *ACS synthetic biology*, vol. 8, no. 7, pp. 1560–1563, 2018.

[37] A. A. Nielsen, B. S. Der, J. Shin, P. Vaidyanathan, V. Paralanov, E. A. Strychalski, D. Ross, D. Densmore, and C. A. Voigt, "Genetic circuit design automation," *Science*, vol. 352, no. 6281, 2016.

[38] T.-C. Liang, Z. Zhong, Y. Bigdeli, T.-Y. Ho, K. Chakrabarty, and R. Fair, "Adaptive droplet routing in digital microfluidic biochips using deep reinforcement learning," in *International Conference on Machine Learning*, PMLR, 2020, pp. 6050–6060.

[39] N. Bonzanni, K. A. Feenstra, W. Fokkink, and E. Krepska, "What can formal methods bring to systems biology?" In *International Symposium on Formal Methods*, Springer, 2009, pp. 16–22.

[40] J. Fisher and T. A. Henzinger, "Executable cell biology," *Nature biotechnology*, vol. 25, no. 11, pp. 1239–1249, 2007.

[41] A. Regev and E. Shapiro, "Cells as computation," in *International Conference on Computational Methods in Systems Biology*, Springer, 2003, pp. 1–3.

[42] M. Grieb, A. Burkovski, J. E. Sträng, J. M. Kraus, A. Groß, G. Palm, M. Kühl, and H. A. Kestler, "Predicting variabilities in cardiac gene expression with a boolean network incorporating uncertainty," *PloS one*, vol. 10, no. 7, e0131832, 2015.

[43] M. A. Inlay, D. Bhattacharya, D. Sahoo, T. Serwold, J. Seita, H. Karsunky, S. K. Plevritis, D. L. Dill, and I. L. Weissman, "Ly6d marks the earliest stage of b-cell specification and identifies the branchpoint between B-cell and T-cell development," *Genes & development*, vol. 23, no. 20, pp. 2376–2381, 2009.

[44] D. Sahoo, J. Seita, D. Bhattacharya, M. A. Inlay, I. L. Weissman, S. K. Plevritis, and D. L. Dill, "MiDReG: A method of mining developmentally regulated genes using boolean implications," *Proceedings of the National Academy of Sciences*, vol. 107, no. 13, pp. 5732–5737, 2010.

[45] D. Sahoo, "The power of boolean implication networks," *Frontiers in physiology*, vol. 3, p. 276, 2012.

[46] A. Yachie-Kinoshita, K. Onishi, J. Ostblom, M. A. Langley, E. Posfai, J. Rossant, and P. W. Zandstra, "Modeling signaling-dependent pluripotency with boolean logic to predict cell fate transitions," *Molecular systems biology*, vol. 14, no. 1, e7952, 2018.

[47] P. Dalerba, D. Sahoo, and M. F. Clarke, "CDX2 as a prognostic biomarker in colon cancer," *The New England journal of medicine*, vol. 374, no. 22, p. 2184, 2016.

[48] J. W. Scannell, A. Blanckley, H. Boldon, and B. Warrington, "Diagnosing the decline in pharmaceutical R&D efficiency," *Nature reviews Drug discovery*, vol. 11, no. 3, pp. 191–200, 2012.

[49] A. Zhavoronkov, Y. A. Ivanenkov, A. Aliper, M. S. Veselov, V. A. Aladinskiy, A. V. Aladinskaya, V. A. Terentiev, D. A. Polykovskiy, M. D. Kuznetsov, A. Asadulaev, *et al.*, "Deep learning enables rapid identification of potent DDR1 kinase inhibitors," *Nature biotechnology*, vol. 37, no. 9, pp. 1038–1040, 2019.

[50] N. J. Gesmundo, B. Sauvagnat, P. J. Curran, M. P. Richards, C. L. Andrews, P. J. Dandliker, and T. Cernak, "Nanoscale synthesis and affinity ranking," *Nature*, vol. 557, no. 7704, pp. 228–232, 2018.

Chapter 8

Education and Workforce Training

The size, reach, and impact of design and design automation have grown significantly over the years, powered by the intellectual efforts of highly trained engineers/researchers who have decades of experience in EDA research and development. However, the entering pipeline is notably shallow, with far fewer new students (at both the undergraduate and graduate level) choosing the technology/circuit/architecture design and design automation profession. In general, undergraduate enrollment in Computer-Engineering and Electrical Engineering has been declining across universities in the United States. It is imperative that these declining numbers be reversed to prevent a further decline in critical onshore IC manufacturing and design capabilities. Several possibilities exist: (i) fewer students find semiconductor and related jobs compelling, ie., there is a perception that these jobs are moving away from the US, (ii) high-school and early-undergraduate students migrating away from these disciplines due to a perception of *software can do more*, (iii) outdated curricula that do not excite students, and (iv) not enough young faculty and role models. Below, a more detailed discussion on the current status, challenges, and needs related to education and workforce developments is provided.

8.1 Core EDA

Period of early EDA excitement. The Mead and Conway VLSI revolution [1] in the early 80's created tremendous excitement in the academic EDA community. The early 80's and 90's were the golden era for academic and industrial EDA: the combination of a simple set of design rules, coupled with structured, hierarchical design abstractions (Transistor $\rightarrow$ Circuit $\rightarrow$ Logic $\rightarrow$ RTL $\rightarrow$ Algorithmic Behavior) created a level playing field for computer scientists to collaborate fruitfully with circuit designers and electrical engineers, resulting in the development of many sophisticated optimization and synthesis algorithms, as well as complex simulation frameworks that enabled what-if analyses for early design space exploration of design alternatives, and rapid concept-to-design cycles. This resulted in a wealth of academic research and tools for creating increasingly complex VLSI chips, in EDA niches such as circuit design [2], physical design [3]; synthesis tools and design flows at the logic [4], RTL and behavioral levels [5]; formal models and equivalence checking to ensure correctness of designs generated by EDA tools [6]; testing and validation of VLSI circuits [7]; etc. Industry and academia collaborated actively during this period, with the annual Design Automation Conference (DAC) [8] drawing thousands of academic researchers and practitioners from a diverse set of small, medium, and large EDA companies. Indeed, the Silicon Valley start-up booms in the late 90's also generated tremendous interest for academics to be active in start-up EDA companies. All of this excitement resulted in great interest for educating students, both at the undergraduate and graduate levels, resulting in the proliferation of many EDA courses and curricula across the country, as well as a strong pipeline of EDA professionals.

Maturation of early academic EDA research. At some level, the early EDA academic and research communities became victims of their own successes in the 2000 decade, with academic and EDA research tools transitioned to the EDA companies and large chip design houses (Intel, AMD, IBM, TI, etc.). Increasingly complex and rapid device

technology advances coupled with competition/secrecy in the industry with regard to advanced technology nodes and design drivers posed significant barriers for academia to know the real challenges faced by designers, and sample data sets that could be used to drive impactful academic EDA innovations. Furthermore, the EDA industry underwent a significant consolidation into a few, large EDA companies, which created further barriers for cooperation between the EDA industry, design houses, and academic research.

The lure of big data, AI/ML, and startups. The 2010 decade opened the floodgates for excitement and innovations in big data analytics and AI/ML. Many EDA-trained students were swept up, both by generous salaries/stock options, as well as overall excitement in these emerging fields. Government funding for EDA did not grow sufficiently to sustain the growing need for research to meet new challenges arising from advanced technologies, newer applications, and increased design complexity. This resulted in many EDA faculty reorienting their research skills in emerging non-EDA arenas, as well as fewer students pursuing EDA research and careers in EDA.

Looking forward. We believe it is critical to reinvigorate the excitement of EDA as a vibrant and critical field. This requires cultivation of seed corn for growth of this field through education of the next generation of EDA students and researchers, through a multi-pronged approach:

Creating new EDA "AlphaGo" moments to ignite excitement. EDA community had multiple such AlphaGo moments in the past, such as outperforming highly trained VLSI designers in circuit layout, automatic generation of RTL code for complex accelerator designs, and industry-wide adoption of EDA techniques for test and verification to replace manual approaches that dominated a large part of the 20th century. However, the EDA community needs to do a better job in communicating such exciting progress to the general public and key stakeholders of NSF, and to energize and excite students and young researchers about the impacts of EDA. Emerging applications can be used as drivers for EDA research to motivate students by demonstrating how EDA is a key enabler in generating impactful outcomes, while providing intellectually stimulating research challenges. Educators and researchers need to look beyond high-volume application spaces to exploit a diversity of applications and emerging technologies that create new opportunities for impact, as well as the potential for exciting system startups. EDA contests and benchmarks suites can generate further interest for students to participate in friendly EDA competitions that could be promoted at premier EDA conferences, and through professional EDA organizations (ACM SIGDA and IEEE CEDA). Of course this requires the analysis of real applications, extraction of key application and technology drivers, and careful balancing of the right abstractions to avoid excessive complexity so that the resulting contests and benchmarks are representatively realistic and challenging, without overwhelming the students.

Culture of openness and sharing. A related, and important goal is to create a culture of openness, sharing, and reproducibility. Students should be trained to think about creating artifacts that are usable and experiments that are reproducible, focusing on applications and technologies that are relevant, and which have the potential for translation into real-world impact. This critically relies on community development of open/shared repositories of data sets, and open-source code for published algorithms, tools and frameworks, as exemplified by successful open-source, community-driven efforts in data analytics (e.g., Apache Spark [9]) and machine learning (e.g., Caffe [10]). EDA faculty need to involve undergraduates in the creation of these data sets and codes, and thus generating a sense of ownership for open-source efforts through co-authorship in publications and presentations.

EDA and CS/Engineering Curricula. EDA has shown tremendous success in engineering complex chips from abstract specifications, using fundamental notions of abstraction, hierarchy, multi-objective optimization, correctness, simulation, design space exploration, etc. Although some of these topics may be covered within different CS and Engineering curricula, as well as at different stages of their education, we believe that many EDA principles are foundational for a strong CS/Engineering curriculum, and therefore should be introduced in basic CS & Engineering education. Furthermore, many EDA-specific courses and curricula are currently focused at the graduate level; we

recommend exposing undergraduate students to exciting EDA problems they haven't seen before, and engage them in EDA challenges and contests. While many CS curricula may not expose students to hardware design, EDA enables a software-centric path for CS students to generate interesting hardware using domain-specific languages (DSLs) and hardware prototyping using FPGAs. Recent DSL compiler frameworks for FPGA accelerators (e.g., for machine learning [11], [12] and image processing [13]) allow CS students to engage in hardware design, and expose them to the vibrancy of the EDA field. These efforts should create a larger pipeline of both Engineering as well as CS students pursuing careers in EDA.

Messaging EDA vibrancy, impact, and stability. The EDA community needs to develop a compelling messaging strategy for young students that emphasizes the vibrancy, impact and stability of EDA as a profession. EDA is a vibrant field, adapting and rising to meet new challenges arising from emerging applications and technologies. EDA is a critical enabler for the specification, design, and development of highly complex electronic systems that fuel the information economy. And EDA as a profession promises stability and job satisfaction, unlike many start-ups in other domains. The role of professional societies such as ACM SIGDA and IEEE CEDA are critical in this regard.

8.2 Beyond Core EDA: Circuit and Systems Design and General Design Automation

Design and design automation go hand-in-hand. Circuits and systems designers are the users of EDA tools while EDA tool development must respond to the needs of new foundational technologies and new applications (including ML/AI/BI applications). Furthermore, EDA principles and solutions can benefit problem domains beyond traditional semiconductor integrated circuits. All these bring new challenges and opportunities.

With the conventional CMOS scaling reaching its limit, close interactions between technology and EDA tool development are indispensable. The rapidly moving ML/AI/BI field further increases the needs of connecting device technologies $\rightarrow$ circuits $\rightarrow$ architectures$\rightarrow$ systems$\rightarrow$ applications. The fields of foundational technologies and NanoSystems for emerging applications create exciting opportunities for students to make meaningful impact. At the same time, the fields present a number of challenges in attracting undergraduate and graduate students:

- It is important to understand the interplay between device technologies, circuits, architectures and applications – a "cross-layer" approach. In contrast, past activities in foundational technologies mostly focused on a single layer (or a few adjacent layers) – mostly materials and devices. While the cross-layer approach is exciting, it is also challenging to create a practical curriculum that covers both depth and breadth sufficiently.
- Access to latest foundational technologies is often limited to a few research groups and even more limited for classroom teaching for various reasons, such as export control challenges, legal IP issues, and competitive commercial nature of these technologies. The gap between industry and academic technology access has only grown. Such limited access severely limits innovations by university students and researchers. Very few researchers at the circuit and architecture levels have an opportunity to create new NanoSystems by exploiting foundational technologies.
- The lack of infrastructure and support for student designs results in very few students getting the opportunity to even design in modern processes. This limits the knowledge of IC design to very few students. It may also reduce the attractiveness of circuit and system design courses to students.
- Similar to the above points, due to the lack of hardware prototyping facilities, very few researchers at the foundational technologies level get the opportunity to create medium- or large-scale hardware demonstrations using their technologies.
- Unlike many other fields (e.g., those related to the development of application software and algorithms), there can be a long cycle to obtain results and to reach gratification, sometimes spanning several years.

- Lots of developments in foundational technologies are happening outside the US (e.g., in China, Europe, S. Korea, and Taiwan). Hence, there are less incentives for students from those regions to pursue research in these fields in the USA. This has a direct consequence on the future student and workforce pipelines in these fields in the US.
- Overly simplistic messages (frequently driven by commercial motives) equating the miniaturization wall or the power wall with the end of hardware technology advances often demotivate young students from entering the field (especially in the US).

Addressing the above challenges require efforts in multiple fronts: from well thought-through messaging and curriculum development to infrastructure support and collaboration among government, industry, and academia. The ideas outlined in Chapter 8.1 are equally applicable here. Effectively dealing with the disciplinary barriers in education (e.g., by forming design institutes) is important in educating engineers/researchers who can be truly creative in the fast moving design and design automation field. To provide a level playing field, the design and design automation community can take inspiration from the open source nature of machine learning development.

System-on-chip (SoC) design and system-level integration skills have emerged as a critical requirement for today's workforce. SoC design and system integration are key to developing hardware-based system solutions that can effectively and efficiently address the requirements of today's complex and multi-faceted applications with varying design specifications and demanding performance requirements. Unfortunately, a majority of academic institutions of higher education are not equipped with the resources, know-how, and tools needed to train such a workforce. These institutions are very good at providing deep technical knowledge of a given field (such as networking, computing, signal processing or communication systems) but fall short when it comes to training a skilled person in the art of combining various point solutions in computing, communication, applications, etc. into a unified hardware-software platform that addresses the applications' needs.

As discussed in Chapter 7, EDA principles and solutions have been applied in recent years to problem domains outside of traditional semiconductor integrated circuits. For example, EDA research has broadened to encompass topics in other fields such as systems biology, lab-on-chip, smart grid, quantum computing, hardware security, AI accelerators, and CPS. There is clearly a need for innovations in education that can prepare the next generation of researchers and practitioners for the new EDA landscape. The traditional curriculum has emphasized semiconductor electronics, chip design, algorithms and formal methods, software engineering, optimization techniques. Future innovations in curriculum design must go beyond these topics and encompass AI/ML, statistics, data science, physics of new types of devices, and the convergence with the life sciences (microbiology and biochemistry). The key is to abstract out EDA concepts that are presented narrowly in the context of chip design, and present them in a broader context so that students can apply these concepts to new domains. In addition, there is an opportunity to integrate EDA concepts in the lifelong learning of working professionals in all these domains, e.g., through training workshops, tutorials, and online courses.

Computing and engineering need to do a better job of training students to be comfortable in multiple disciplines. EDA is traditionally a model for this activity—the field requires expertise in combinatorial algorithms, numerical algorithms, and circuits/devices. EDA is a natural hub for the next step in multidisciplinary design. Purely analytical methods used in traditional engineering are no longer sufficient to meet the complex requirements of modern systems—simulation is required. Complex design spaces are too large to explore by manual design—synthesis and optimization are required. Large, complex systems cannot be verified manually—computer-aided verification methods are required.

The EDA research community can play an important role in education and workforce training to enable next-generation system design. The EDA community can develop educational materials for this broader definition of system design and design automation. Tools can be developed to support new types of design—educational tools do not need to

support everything required for industrial adoption. Benchmarks, data sets, and sample designs can be developed to enhance learning.

EDA education should continue to emphasize fundamental principles, not just techniques. In the age of AI, it is critical to educate students in learning when to use which: physical modeling or AI. EDA education should also help train practitioners in the domains where ad hoc design techniques are traditionally used so that new EDA tools and methodologies can be more widely adopted.

Bibliography

[1] C. Mead and L. Conway, *Introduction to VLSI systems*, 1980.

[2] ISSCC, *International solid-state circuits conference (ISSCC)*. [Online]. Available: http://isscc.org/.

[3] ISPD. [Online]. Available: http://www.ispd.cc/.

[4] IWLS. [Online]. Available: http://www.iwls.org/.

[5] ESWEEK. [Online]. Available: https://esweek.org/.

[6] CAV. [Online]. Available: http://i-cav.org/.

[7] ITC. [Online]. Available: http://www.itctestweek.org/.

[8] DAC. [Online]. Available: https://www.dac.com/.

[9] M. Zaharia, R. S. Xin, P. Wendell, T. Das, M. Armbrust, A. Dave, X. Meng, J. Rosen, S. Venkataraman, M. J. Franklin, *et al.*, "Apache spark: A unified engine for big data processing," *Communications of the ACM*, vol. 59, no. 11, pp. 56–65, 2016.

[10] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proceedings of the 22nd ACM international conference on Multimedia*, 2014, pp. 675–678.

[11] T. Moreau, T. Chen, L. Vega, J. Roesch, E. Yan, L. Zheng, J. Fromm, Z. Jiang, L. Ceze, C. Guestrin, *et al.*, "A hardware–software blueprint for flexible deep learning specialization," *IEEE Micro*, vol. 39, no. 5, pp. 8–16, 2019.

[12] A. Sohrabizadeh, J. Wang, and J. Cong, "End-to-end optimization of deep learning applications," in *Proceedings of the 2020 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, 2020, pp. 133–139.

[13] J. Li, Y. Chi, and J. Cong, "HeteroHalide: From image processing DSL to efficient FPGA acceleration," in *Proceedings of the 2020 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, 2020, pp. 51–57.

Chapter 9

Recommendations to NSF

Based on the discussions in this report, we make the following recommendations.

9.1 Raise Awareness

Recommendation and Expected Outcomes

- There is an immediate need for the NSF to help organize and coordinate federal/state-level awareness campaigns, at least at the levels of artificial intelligence, robotics and quantum computing campaigns, to emphasize: (a) the critical importance and tremendous potential of hardware technologies and NanoSystems to revolutionize almost every aspect of all our lives; and, (b) the increasingly crucial role of EDA and its growing opportunities in directly impacting hardware and software technologies moving forward.
- The NSF should support the EDA community in creating large-scale (both in terms of problem complexity and participating teams) competitions/challenges to ignite the interest of students and young researchers in NanoSystems design and EDA.

Justification

Chapter 8 provides extensive justification for this recommendation. It is essential that the NSF raises awareness at the federal, state and local levels regarding the criticality of NanoSystems design and design automation, and foundational technologies in advancing the future of computing, communications and information technology:

- Without such advances, our dreams about advancing AI, communication (6G and beyond) and quantum computing will not be realized.
- The field of NanoSystems design and design automation, and foundational technologies is rich with many promising research ideas that can advance the performance, energy efficiency and scalability of computing significantly by orders of magnitude moving forward. The perception about the end of foundational technologies as a consequence of the power wall and the miniaturization wall is misguided.
- Overly simplistic messages (frequently driven by commercial motives) equating the miniaturization wall or the power wall with the end of hardware technology advances often demotivate young students from entering the field (especially in the US).
- US competitors (such as China, the European Union) are investing heavily in the domains of foundational technologies, NanoSystems and EDA. To ensure economic competitiveness, technology leadership and national security, the US must invest heavily in this area, at least at the scale of quantum computing.
- There is an urgent need for a redoubled effort to focus NanoSystems research spanning devices, design, architectures, EDA, and manufacturing beyond conventional silicon CMOS.

The second of these recommendations implies, and goes beyond, the need to provide students and future generations of researchers greater access to EDA tools and NanoSystems fabrication infrastructure (Chapter 9.2), to

ensure that these infrastructure resources will be put to great use in the long term to maintain and strengthen the competitiveness of the nation's human resources in the global EDA and NanoSystems marketplace.

NSF CISE, especially its Computing and Communication Foundations Division, is in the ideal position for this purpose given its focus on Software and Hardware Foundations (SHF) connecting hardware technologies to software applications.

9.2 Infrastructure

9.2.1 Technology Access for Design and EDA of NanoSystems

Recommendation and Expected Outcomes

The NSF should facilitate access to industrially-offered technologies:

- Advanced silicon-CMOS technologies (e.g., 5nm and beyond).
- Beyond silicon-CMOS technologies (e.g., new logic, memory and integration technologies).

These include technologies offered by prominent foundries (e.g., Global Foundries, Intel, Samsung, TSMC) as well as industrial facilities creating special technologies (e.g., logic technologies such as carbon nanotube FETs and ferroelectric FETs, memory technologies such as FeRAM, MRAM, PCRAM and RRAM, integration technologies such as bonding, monolithic 3D, TSV 3D and other packaging approaches, interconnect technologies such as photonics, power technologies such as GaAs, GaN and SiC). In addition to commercial entities, access to research facilities (including international ones such as CEA LETI in France, IMEC in Belgium, Fraunhofer/Leibnitz Institutes in Germany, ITRI and TSRI in Taiwan) should also be seriously considered.

Justification

Currently, such access is limited to only a select group of researchers in the U.S. (through their personal connections or through specific programs funded by agencies such as DARPA and IARPA). A much broader set of U.S. researchers (especially in the domains of circuit design, architecture and EDA) must be able to gain such access to ensure:

- A robust workforce pipeline by attracting students to work in these fields.
- Competitiveness of U.S. graduates for employment at leading-edge companies.
- Competitiveness of U.S. companies as well as new startups from U.S. research.
- Competitiveness of U.S. research and innovation in NanoSystems for economic growth.
- Competitiveness of U.S. research and innovation in EDA uniquely spurred by access to advanced silicon CMOS technologies **and** new logic, memory and integration technologies beyond traditional silicon CMOS.

Several reports have emphasized this need. A few examples are given below:

- 2021 IEEE VLSI Circuits and Systems Letter, Article on Semiconductor Microelectronic Research at NSF/CISE (https://ieeecs-media.computer.org/tc-media/sites/18/2021/05/13171826/VCaSL_May2021-newsletter.pdf)
- 2020 NSF Foundry Meeting report (https://nsfedaworkshop.nd.edu/assets/429148/nsf20_foundry_meeting_report.pdf)
- 2017 PCAST report on Ensuring Long-Term U.S. Leadership in Semiconductors (https://obamawhitehouse.archives.gov/sites/default/files/microsites/ostp/PCAST/pcast_ensuring_long-term_us_leadership_in_semiconductors.pdf)

Moreover, research groups in other countries (especially China, Taiwan, and Europe to some extent) enjoy such access resulting in a competitive edge compared to researchers in the US.

To orchestrate streamlined access to the above technologies, a new MOSIS-like effort (with far more expanded portfolio of technologies) may be suitable. While jointly funded efforts between multiple U.S. government agencies may seem natural, the long-term nature of such accesses might make the NSF a natural home for such activities. For example, a 10-year access program (with some agility to make changes to accessible technologies every 5 years) may be suitable for the NSF compared to other agencies that often focus on specific projects. U.S. industries offering and using such technologies may need to get involved. At the same time, it is important to balance the long-term research vision behind such accesses vs. short-term industrial product goals. A few challenges are highlighted:

- What technologies and fabrication facilities to focus on?
- Hardware prototypes for exciting NanoSystems might involve heterogeneous technologies offered by disjoint fabrication facilities. How to coordinate successful tapeouts in such situations?
- Leading-edge commercial foundries often limit technology access to hardware tapeouts only. How can EDA researchers obtain access to a wide variety of (leading) technologies?
- What is the role for open-source development in driving EDA and NanoSystems innovation while remaining in touch with what is supported for fabrication in commercially available technologies?

9.2.2 Fabrication and Design Support for Exploratory NanoSystems

Recommendation and Expected Outcomes

- NSF should help establish **and** support facilities for prototyping medium- to large-scale NanoSystems, beyond a few (1 to 1,000) stand-alone devices as is common today. This might take the shape of **(a)** creating new hardware fabrication facilities or **(b)** expanding capabilities of "exploratory" fabs. The key is to ensure that such facilities can "quickly" customize their fabrication lines to implement novel technology ideas (envisioned and created by researchers in their university laboratories) for realizing hardware prototypes of exploratory NanoSystems.
  - Beyond traditional focus on transistor, memory and sensing technologies, it is crucial to explore innovative integration and thermal technologies as well.
- NSF should enable academic researchers to access such facilities to realize their nanotechnology ideas and translate them into working NanoSystems.
- It is often valuable to build NanoSystems demonstrations on top of existing silicon infrastructure, e.g., new nanotechnologies integrated on top of silicon wafers to demonstrate interesting circuit- and system-level capabilities – a "sauce over pasta" (or "curry over rice") approach. In that case, NSF should enable access to such silicon wafers from industrial fabrication facilities.
- Design enablement through EDA that is essential for enabling designers to create new circuits and architectures using new technologies. Three important aspects are:
  - Creation of and access to PDKs and libraries, design tools and flows to enable new hardware prototypes.
  - There must be tight integration between new technology development and new EDA for the following reasons:
    - New EDA tools must translate application needs (e.g., energy, throughput, security) into technology targets (e.g., improvements in logic, memory, connectivity) that will guide technology researchers.
    - EDA acts as a technology enabler (as articulated in Chapter 3) to unlock potential benefits of new technologies.
  - Enable **correct** technology benchmarking, a major challenge (discussed in Chapter 4).

Justification

The computing needs of the coming generations of applications (including ML/AI, 5G/6G communication, quantum computing) are unlikely to be met by isolated "business as usual" improvements in technology, circuits and

architectures. Fortunately, there are many promising ideas at the level of nanotechnologies (logic and memory devices, integration technologies, thermal solutions) and also at the level of NanoSystems that leverage the unique properties of such foundational technologies to create new and transformative architectures. The combination of new nanotechnologies and new NanoSystem architectures promises to deliver large benefits (functionality, throughput, energy efficiency) of future computing systems.

At this exact moment, it is practically impossible to realize new NanoSystems concepts at university facilities (except some heroic efforts by a handful of researchers). Very little exists in terms of design enablement for new nanotechnologies. Likewise, research in design and EDA of computing systems is mostly confined to technology offerings by mainstream commercial foundries. This has resulted in a research culture which encourages focus on hardware technologies that are offered by mainstream foundries. There is little incentive to explore new hardware technologies not offered industrially. It is difficult for commercial fabrication facilities to pursue "high-risk" approaches (with potentially high payoffs). This creates vicious cycle which stifles innovations particularly at a critical point in time when business as usual is no longer viable. To overcome this challenge, U.S. competitors are already investing heavily (e.g., CEA LETI and IMEC in Europe, ITRI in Taiwan).

Thus, fabrication of exploratory NanoSystems is crucial to ensure:

- Competitiveness of U.S. graduates and a robust workforce pipeline.
- Competitiveness of U.S. companies specializing fabrication technologies, hardware systems, fabrication equipment and EDA, as well as new startups from U.S. research.
- Competitiveness of U.S. research and innovation in computing which is critical for economic growth and prosperity. As the past two decades (2000-2020) have shown, loss of competitive edge in foundational technologies often results in a loss of competitive edge in hardware systems (and software systems as well in an era when software companies are building hardware customized to their applications).

Translating academic technology ideas into working technologies (at the facilities) and NanoSystems prototypes requires new programs coordinated by the NSF.

Access to silicon wafers from leading-edge foundries for *sauce-over-pasta* (or *curry-over-rice*) NanoSystems demonstrations may be possible through a coordinated effort similar to Section 9.2.1. The facilities themselves will need to be involved since a lot will depend on the fabrication tools supported.

Obviously, such an ambitious infrastructure won't happen instantaneously. In the meantime, technologies created at exploratory fabs and custom foundries in the US and also in research facilities (e.g., CEA LETI, IMEC) can be leveraged.

9.2.3 Community Infrastructure for Design Enablement

Recommendation and Expected Outcomes

NSF should help establish and support community-wide design infrastructure (preferably in the cloud) with both industrial-strength tools and open-source research EDA tools for:

- Design enablement for NanoSystems, including PDK and library development and design flows (high-level synthesis, logic synthesis, timing/power/reliability/thermal analysis and physical design).
- Design verification of IP blocks and beyond.
- Test, reliability and security features.

Justification

We already discussed the need for design enablement for NanoSystems in Chapter 4.3. In addition, 21st-century hardware designs resort to staggering complexity to meet system functionality, performance, and energy objectives

(as has been highlighted in many publications). For example, hardware accelerators have become crucial in building energy-efficient (heterogeneous) System-on-Chips (*SoCs*). Unlike general-purpose processors, a wide variety of heterogeneous SoCs must be designed quickly by small design teams.

Beyond design, existing validation and test methods barely cope with today's complexity – as a result, hardware design bugs and defects are inexorably rising. Several reliability failure mechanisms, largely benign in the past, are becoming visible at the system level. A large class of future systems will require tolerance of hardware errors during their operation. Recent studies by Facebook and Google indicate significant degrees of errors causing silent data corruption in the cloud. Security concerns (from supply chain challenges to side-channel attacks) are also growing. New approaches are essential to prevent incorrect system operation, security risks, and expensive product delays and recalls.

The EDA industry narrowly focuses on very near-term approaches and there is very little support by the EDA industry for long-term research (both support for academic research as well as research inside EDA companies). Opaque EDA tools provide little opportunity for research advances. As a result, interest in EDA research is declining in the US at a time when new EDA breakthroughs are crucial for the future of the semiconductor industry. To put into perspective, interest in EDA research is skyrocketing among US competitors such as China and Taiwan.

New thriving community-based infrastructure (similar to nanoHUB infrastructure by the NSF, open-source compiler infrastructure in software systems such as LLVM, open-source Boolean Satistfiability and SMT solvers in computer science) is essential to break this trend and ensure:

- A robust workforce pipeline for the next generation of EDA without which the US semiconductor industry will lag significantly behind competitors.
- Competitiveness of the U.S. semiconductor industry in general (and the US EDA industry in particular).
- Competitiveness of U.S. research and innovation in computing systems. U.S. competitors are significantly investing in this domain while the interest among U.S. researchers in waning for reasons discussed above.

It is important to provide well-supported design flows using both industrial-strength tool flows and open-source research tool flows. The former targets researchers building complex VLSI designs who need timely support. The latter is crucial for researchers developing new EDA algorithms/tools allowing them to demonstrate the practicality and effectiveness of their approaches.

NSF, with its deep experience with nanoHUB, is uniquely positioned to drive a new program on this topic. DARPA, which initiated the Posh Open-Source Software and Hardwware (POSH) program, might be interested in playing a role. The scope is bigger than nanoHUB, and the three prongs discussed above (design, verification, test/reliability/security) are crucial.

9.3 Fundamental Research Topics:

9.3.1 New topics on Traditional EDA

Recommendation and Expected Outcomes
NSF should initiate new research programs focusing on

- New EDA approaches to address massive complexity at all stages of design, test and in-field operation.
- New EDA approaches for new families of systems enabled by a wide variety of new 2.5D and 3D integration technologies.
- Special emphasis on robust operation with resilience to bugs, manufacturing defects, reliability failures and security attacks.

- New EDA approaches to facilitate NanoSystem designs based on emerging logic, memory and integration technologies.
- New formulation of EDA problems based on theoretical foundations in optimization and machine learning (ML). In particular, given recent interest in the use of machine learning for EDA, NSF should consider setting up special programs to encourage interactions and collaborations between EDA researchers and ML experts to jointly address the challenges of ever-increasing design automation challenges.

Justification

We have provided justification in Section 9.2.3 and also in Chapter 3 and Chapter 4.

9.3.2 EDA Beyond Hardware Platform Creation

Recommendation and Expected Outcomes

NSF should initiate new research program on new EDA approaches to software productivity on heterogeneous hardware platforms for existing / new domains.

Justification

As discussed in Chapter 3, for heterogeneous and accelerator-rich computing systems with a wide variety of accelerators and general-purpose processors, programmers must navigate a large design and optimization space. Moreover, accelerators require programmers to manage many concerns explicitly in software. This problem gets even more complex as accelerators evolve rapidly. Hence, it is crucial to support quick (days instead of many months) bring-up of software stacks that can adapt to a moving targets. Most techniques used by the EDA community (synthesis, mapping, placement, routing, and verification) are crucial. By extending EDA beyond hardware platform creation, EDA benefits can reach not only tens of thousands of hardware designers, but also millions of software programmers and even potentially data scientists as well.

9.3.3 Codesign for NanoSystems

Recommendation and Expected Outcomes

There is an immediate need for new research programs focusing on Co-design for NanoSystems, connecting hardware circuits and architectures with applications on one end of the spectrum and foundational nanotechnologies on the other – a co-design approach. Three examples are given below:

- Connect abundant-data workloads (e.g., speech and video processing, graph processing, data analytics, security) with new nanotechnologies.
- Connect the wide variety of (existing and new) ML/AI models with new nanotechnologies.
- Connect emerging models of computation (stochastic computing and p-bits, approximate computing, Ising and others) with new nanotechnologies to realize a wide variety of NanoSystems (including digital, analog-heavy, low-temperature, superconducting, coupled oscillators, thermodynamic and other implementations).

We expect some hardware demonstrations to be part of such co-design efforts.

Justification

As discussed in Chapter 4, 21st-century computing systems are characterized by a wide diversity of applications, algorithms and hardware architectures that are changing rapidly. Similarly, an explosion of new concepts in *foundational* nanotechnologies and NanoSystems is also emerging. There is growing recognition about combining these wide variety of technologies in innovative ways to create new architectures optimized for various application domains. Such approaches require a new set of EDA tools, different from current commercial offerings. This creates the need for co-design across technology, architecture and application levels. Here is a sample of a few such co-design questions:

- Given a set of tasks from an application domain and a set of foundational technologies (e.g., for logic, memory and connectivity/integration), how do we jointly explore the space of applications (e.g., adaptation at the edge, implants and brain-computer interfaces, robotics, applications involving sensing, decision-making and actuation), algorithms, architectures and technologies that achieve the best possible application-level energy and execution times?
- How do we translate application-level needs (e.g., energy, throughput, latency) into technology-level targets (e.g., logic energy/speed/density, memory energy/speed/density, density of connections) and derive (new) technologies that meet these targets?
- Can circuit-, architecture- or application-level techniques overcome inherent imperfections, variations or reliability challenges associated with various foundational technologies?
- How do we address thermal challenges for coming generations of 3D integration technologies?

Such unprecedented technology-architecture-application affinity creates unique opportunities: (1) innovative EDA approaches as technology enabler (Chapter 3) not only with respect to classical metrics (energy, throughput, cost) but also emerging metrics (e.g., security, privacy, accuracy of results, robustness to manufacturing and environmental variations); and, (2) new benchmarking opportunities for foundational nanotechnologies and NanoSystems (see Chapter 4).

The scope of such a program must be deeply analyzed – given the inter-disciplinary aspect, NSF CISE is expected to be at the forefront (and a driver) of such activities. Synergies with design enablement activities in Section 9.2.3 are expected.

9.3.4 NanoSystems Hardware Prototypes

Recommendation and Expected Outcomes
NSF should initiate new research programs focusing on:

- Establishment of new nanotechnologies in exploratory fabs (deeply connected with Section 9.2.2, but at the same time driven by co-design efforts in Section 9.3.3).
- Demonstration of medium- to large-scale hardware prototypes for co-designed NanoSystems using the nanotechnologies established (by leveraging the infrastructures in Section 9.2.1 and Section 9.2.3).

These programs are expected to be major efforts with **high costs**, **high risks** and **high rewards**, tightly managed. Each such project is expected to be at a focused effort at a 5-year scale extendible to another 5 years with very tight control. Adaptivity is critical as various factors can change in the course of such ambitious projects. The DARPA 3DSoC program can act as an initial working model.

Justification
Sections 9.2.1,9.2.2, and 9.3.3 provide justification.

Given the scope, multiple agencies are expected to be involved. Considerations beyond the scope of this document are required to formulate such programs.

9.3.5 EDA for Machine Learning and Machine Learning for EDA

Recommendation and Expected Outcomes
NSF should initiate new research programs specifically focusing on the following:

- Co-design for machine learning/artificial intelligence/bio-inspired (ML/AI/BI) systems, across technology, circuit, architecture, algorithm, system, and application levels.
- Medium- and large-scale hardware and system prototyping of ML/AI/BI systems.

- Community shared, very large hardware-software infrastructure for ML/AI/BI research at scale.
- Cross-disciplinary co-exploration of ML/AI/BI systems, especially between computer science/engineering, cognitive science, and neuroscience.
- Application of ML/AI techniques to boost EDA develpments at larger scale and with greater autonomy.

Justification

As discussed in Chapter 5 , ML/AI/BI hardware is a rapidly growing and highly impactful area that deserves major attention. Hardware paradigms both in conventional CMOS and in emerging technologies accompanied by constraints of their own can be retrofitted to exploit latitudes available in algorithmic considerations, thus giving rise to possibilities of hardware-software-algorithm co-design. Such research can inspire new algorithmic innovations, or could alternatively be driven by empirical or utilitarian considerations. A concomitant issue is how to verify ML/AI/BI hardware, perhaps supplanting known formal or semi-formal methods. New research is also necessary to identify hardware bugs/faults, and methods to mitigate their effect on hardware performance in the context of ML applications.

Breakthrough ML/AI/BI advances benefit from highly interdisciplinary interactions between computer engineers, algorithm designers, EDA tool developers, and increasingly an infusion of cognitive scientists, neuroscientists, and bioengineers in elevating the understanding of how the embodied brain computes and interacts with its environment towards more autonomous, effective, efficient and resilient operation of computing machinery.

Hardware-software co-design for ML/AI/BI at the scale of, say the GPT-3 level of problems, is extremely computationally heavy. It is impractical, and wasteful, to duplicate the commensurate hardware resources in individual investigator laboratories, or multiple collaborative networks. NSF should make an effort to invest in openly shared fully reconfigurable/programmable compute-resources that reach economy of scale in TPU/GPU/FPGA parallel computing hardware to enable efficient and effective design exploration by the community at large. NSF mandates for data-sharing to release models and training/test sets would ensure reproducibility and allow other researchers to build off results. Currently, very few research groups are able to use such co-design techniques.

ML is also emerging as a powerful approach in developing EDA techniques for efficient, reliable and secure hardware design, including better and more efficient validation, verification, and detection of anomalous behavior of the system due to malicious attack. One may also optimize a variety of different system architectures, including for example, those suitable for cloud/edge computing by predicting/characterizing the nature of the computational load, utilize available computing resources, or handle data loss/corruption, and enable federated learning and inference via the use of ML algorithms, thus improving overall system performance.

NSF CISE is ideally positioned to drive this effort, especially in collaboration with other ML/AI and BI initiatives.

9.3.6 EDA for Quantum Computing and Other Emerging Computing Technologies

Recommendation and Expected Outcomes

In collaboration with the National Quantum Computing Initiative, NSF should initiate a new research program on design automation for quantum computing, which supports efficient synthesis and compilation from applications in high-level programming specifications to the family of rapid expanding quantum computing devices, including future domain-specific quantum computing systems.

Justification

Given the steady advances in quantum computing technologies and rapid expansion of quantum computing applications, there is a pressing need to support various domain experts to develop new applications on existing and future quantum systems at a high-level of programming abstraction. However, the existing quantum compilation tools are far away from optimal, even for the moderate complexity of current quantum devices, as discussed in Chapter 3. It will be

highly beneficial to explore the possible extensions of highly successful electronic design automation techniques for complex VLSI systems to quantum computing, and encourage collaborations between the EDA researchers and quantum computing technologists. Such program can also stimulate the application of EDA methodologies and algorithms to enable other emerging computing technologies (such as molecular computing and bio-computing).

NSF CISE should drive this effort, in collaboration with the National Quantum Computing Initiative.

9.4 Disciplined Engineering System Design Automation

Recommendation and Expected Outcomes

NSF should initiate new research programs on design automation for engineering a wide variety of systems for which EDA principles are immediately applicable.

Examples of such systems include autonomous vehicles (cars, drones), networked systems, energy systems, biological and lab-on-a-chip systems discussed in Chapter 5.

Justification

As discussed in Chapter 7, a number of application domains that can benefit tremendously from the systematic design automation processes that have enabled today's electronic systems. The confluence of design automation with these application domains bring new challenges and opportunities.

Given the interdisciplinary nature of such research programs, multiple units across NSF (with CISE being a major player) or even multiple agencies are expected to be involved.

9.5 Education and Workforce Development

Recommendation and Expected Outcomes

- The NSF must create ways to attract high-school and undergraduate students to the critically important field of NanoSystems design and design automation, and foundational technologies to advance future computing. Such efforts are essential for the US but are missing today.
- The NSF should work closely with diversity and inclusion experts to reflect the diversity of the US workforce, create a community of acceptance, and create a community of excitement and innovation around NanoSystems design and design automation, and foundational technologies, which will help attract top diverse candidates to the field.
- Special NSF CISE Fellowships at the undergraduate, masters and PhD levels for students pursuing research in NanoSystems design and design automation, and foundational technologies (with emphasis on diversity as well) can make a tremendously positive impact. The NSF should create ways to lower the entry barrier and shorten the learning curve for students to participate and receive training in this field (including supporting internships in relevant industries).
- Similar to technology access in research (detailed above in Section 9.2.1), the NSF should find ways to incentivize and assist universities to develop and offer engaging courses on NanoSystems design and design automation, and foundational technologiess (with the possibility of taping out exciting nanosystems ideas using nanotechnologies as part of course projects).

Justification

Chapter 8 provide justification.

9.6 Structure of NSF projects

NSF should consider significantly bigger projects at a 10-year scale with obvious intermediate milestones. Such projects can be inspired by multiple objectives such as multidisciplinarity and/or maintaining continuity. Center scale research is perhaps the only known way to NSF to encourage academic researchers to leave their academic siloes and engage in collaborations in neighboring disciplines. Specifically:

1 Existing funding mechanisms for CISE Expeditions in Computing (EiC), Engineering Research Center (ERC), and Science and Technology Center (STC) type projects with large teams are essential to retain critical mass in EDA and NanoSystems design research, and should continue to receive full support going forward.

2 Focused exploration backed by significant funding with the goal of demonstrating hardware prototypes need to be pursued. Such efforts can be coupled with innovative ways of promoting translational research. In the past, NSF ERCs have traditionally played a role in the broader arena of engineering disciplines, but focused efforts on semiconductor microelectronics have been lacking. On the other hand, while some STCs have played important role in exploring the underlying basic science, due to their very nature, translational aspects of research outcome have been largely missing.

3 While serendipitous research resulting in unexpected breakthroughs may have been more common in other areas of science and technology, semiconductor microelectronics has progressed to its present state steadily over several decades of sustained incremental development. Thus, center scale efforts such as the CISE EiC, ERCs or STCs mentioned above for periods of 5-10 years may not be enough for sustained support (e.g., the National Nanotechnology Initiative, has been in existence for two decades, and despite tens of Billions of dollars of expenditure accompanied by notable achievements, its impact on semiconductor microelectronics has been relatively subdued). There do exist, however, examples of longer term support even within the NSF framework, namely the “Mathematics Institutes” supported by the NSF Division of Mathematical science (DMS), several of which have enjoyed more than 20 years of funding. Given the present importance and impact of the field, similar models of support could be explored for semiconductor microelectronics as well.

4 NSF should create mechanisms for funding projects (requiring end-to-end expertise) targeting focused exploration backed by significant funding with the goal of hardware demonstrations. This approach will incentivize high risk system-level demonstrations (with room for failure) over short-term progress indicators alone (such as publications). 10-year projects with intermediate quantitative goals and frequent reviews (e.g., quarterly reviews, major yearly reviews with go/no-go criteria to track progress backed by major funding increments upon success) may be appropriate for such purposes. Industry as funded entities to unlock collaborations with academic teams. Such mechanisms can foster new levels of collaborations between industry and academia.

Appendix A

Workshop Information

A.1 Organizer and Steering Committee

Following are the organizer and steering committee members of the Workshop:

Sankar Basu (Organizer), National Science Foundation
Gert Cauwenberghs, UC San Diego
Jason Cong, UC Los Angeles
X. Sharon Hu, University of Notre Dame
Pinaki Mazumder, National Science Foundation
Subhasish Mitra, Stanford University
Wolfgang Porod, University of Notre Dame

A.2 Workshop Agenda

Below is the complete program of the Workshop. All the plenary speakers and panelists have contributed to the report contents in some form or another. The roundtable panelists have contributed directly to the writing of the corresponding sections.

December 14, 2020

10:45am – 11:00am: Introduction
Sethuraman Panchanathan (NSF)
Margaret Martonosi (NSF)
Sankar Basu (NSF)

11:00am – 1:30pm: EDA tools and methodologies
Plenary talk: Giovanni De Micheli (EPFL): Design should be as simple as possible, but not simpler
Plenary talk: Anirudh Devgan (Cadence): Computational Software and Future of EDA
Panel discussion:
Massoud Pedram (USC): Moderator
Shawn Blanton (CMU)
Andreas Gerstlauer (UT Austin)
Farinaz Kaushanfar (UC San Diego)
Wolfgang Kunz (TU Kaiserslautern)
David Pan (UT Austin)
Jan Rabaey (UC Berkely)

1:30pm – 3:30pm: Break/Poster Presentation
Participated by the recipients of the NSF CAREER Awards.

3:30am – 6:00pm: Foundational technologies
Plenary talk: Tsu Jae King Liu (UC Berkeley): Nano-electro-mechanical switches for future computing paradigms

Plenary talk Plenary talk: Joerg Appenzeller (Purdue): Emerging applications from two-dimensional systems and their potential for 3D heterogeneous integration

Panel discussion:

Vijay Narayanan (PSU): Moderator
Keren Bergman (Columbia)
Sayeef Salahuddin (UC Berkeley)
Dmitri Strukov (UCSB)
Jian-Ping Wang (U of Minnesota)
Philip Wong (Stanford)
Todd Younkin (SRC)

December 15, 2020

10:45am – 11:00am: Introduction

Rance Cleaveland (NSF)
Sankar Basu (NSF)

11:00am – 1:30pm: AI/ML/Brain-inspired hardware design

Plenary talk: Bill Dally, (Nvidia/Stanford): Sustainable Computing via Domain-Specific Architecture and Efficient Circuits

Plenary talk: Kwabena Boahen (Stanford): 3D Silicon Brains

Panel discussion:

Tajana Rosing (UCSD): Moderator
Andreas Andreou (Johns Hopkins)
Deming Chen (UIUC)
Amir Khosrowshahi (Intel)
Rajit Manohar (Yale)
Chris Rowen (Cisco (BabbleLabs))
Kaushik Roy (Purdue)

1:30pm – 3:30pm: Break/Poster Presentation

Participated by the recipients of the NSF medium and/or CRI Awards.

3:30pm – 6:00pm: New application domains

Plenary talk: Alberto Sangiovanni Vincentelli (UC Berkeley): Tools and Abstractions: how to Bring IC-like Design Productivity to Buildings and Artificial Living Organisms

Plenary talk: George Varghese (UCLA): Network Design Automation- When Clarke Meets Cerf

Plenary talk: Supriyo Datta (Purdue): Probabilistic Computing with p-Bits

Plenary talk: Rahul Sarpeshkar (Dartmouth): Rapid and Rational Covid-19 Drug-Cocktail Discovery

Plenary talk: Srini Devadas (MIT): Can Security be Automatic in Computer Systems?

December 16, 2020

10:45am – 11:00am: Introduction

Erwin Gianchandani (NSF)
Sankar Basu (NSF)

11:00pm – 1:00pm: Roundtable discussions

RT1: EDA tools and methodologies

Participants:

- Jason Cong, University of California, Los Angeles (lead)
- Nikil Dutt, University of California, Irvine (co-lead)
- Igor Markov, University of Michigan
- Abhijit Chatterjee, Georgia Tech
- Clark Barrett, Stanford
- Andrew Kahng, UCSD
- Sharad Malik, Princeton
- Zhiru Zhang, Cornell
- Antun Domic, Stanford (ex-Synopsys)
- Giovanni DeMicheli, EPFL

RT2: Foundational technologies

Participants:

- Subhasish Mitra, Stanford University (lead)

- Sayeef Salahuddin, University of California, Berkeley (co-lead)
- Deji Akinwande, University of Texas at Austin
- Joerg Appenzeller, Purdue University
- Pierre-Emmanuel Gaillardon, University of Utah
- Jean Anne Incorvia, University of Texas at Austin
- Jeehwan Kim, MIT
- Vijaykrishnan Narayanan, Pennsylvania State University
- Michael Niemier, University of Notre Dame
- Shaloo Rakheja, University of Illinois at Urbana-Champaign
- Max Shulaker, MIT
- Geoffrey Vaartstra, MIT
- Evelyn Wang, MIT
- H.-S. Philip Wong, Stanford University
- Shimeng Yu, Georgia Tech

RT3: AI/ML/brain-inspired hardware design

Participants:

- Gert Cauwenberghs, University of California, San Diego (lead)
- Edith Beigné, Facebook (co-lead)
- Shantanu Chakrabartty, Washington University in St. Louis
- Song Han, MIT
- Siddharth Joshi, University of Notre Dame
- Tim Mullen, Intheon Inc.
- Bruno Olshausen, University of California, Berkeley
- Priyanka Raina, Stanford

RT4: New application domains

Participants:

- Massoud Pedram, University of South California (lead)
- X. Sharon Hu, University of Notre Dame (co-lead)
- Krishnendu Chakrabarty, Duke University
- Rolf Ernst, TU-BS
- Eby Friedman, University of Rochester
- Leana Golubchik, USC
- Chris Myers, University of Utah
- Boris Murmann, Stanford University
- Debashis Sahoo, UCSD
- Subarna Sinha, Machine Learning Engineering, 23andMe
- Marylin Wolf, University of Nebraska–Lincoln

RT5: Physics-inspired hardware design

Participants:

- Wolfgang Porod, University of Notre Dame (lead)
- Jennifer Hasler, Georgia Institute of Technology (co-lead)
- Andreas Andreou, JHU
- Kerem Çamsari, UCSB
- George Csaba, Pazmany Univ, Hungary
- Todd Hylton, UCSD
- Alex Khitun, UC Riverside
- Jaijeet Roychowdhury, UC Berkeley
- Rahul Sarpeskhar, Dartmouth U
- Zoltan Toroczkai, ND
- Yoshi Yamamoto, Stanford
- Ata Zadehgol, U Idaho

1:00pm – 1:30pm: Summary from roundtable discussions