6.5930/1

Hardware Architectures for Deep Learning

Final Project

March 11, 2026

Joel Emer and Vivienne Sze

Massachusetts Institute of Technology
Electrical Engineering & Computer Science

Design Project

- Project
  - Choose from a list of suggested projects
- Use tools from labs
  - AccelForge, Python
- Teams
  - Two students per team
  - Each will have a TA mentor
- Schedule
  - March 11 – List of projects released
  - March 18 – Submit project selection: https://forms.gle/GjTHg6iHNxRhDeCE9
  - **March 30 – April 29** – Weekly check ins + milestone report outs with mentor during lecture (**attendance at weekly check ins is mandatory**)
    - **April 17 - Milestone 3 proposal due**
  - May 1 – Project Report Due
    - Graduate groups (w/ at least one member enrolled in grad version) include a 1-page overview on related work
    - Poster sessions May 4, 6, and 11 (you will be expected to attend one of the poster sessions to review other projects)

Lecture time dedicated to project after spring break

Design Project

- Design project allows for deeper understanding and application of concepts covered in course
  - Explore more advanced functionality of modeling tools used in the labs

- Projects will have three key milestones
  - Milestone 1 (April 6): Read relevant paper and model prior work (baseline design)
  - Milestone 2 (April 13): Perform various modifications to analyze impact on design metrics. Demonstrate understanding the various tradeoffs.
  - Milestone 3 (April 27): Open-ended design space exploration (Proposal due April 17)

Lecture Time Dedicated to Project after Spring Break

Mar 23 - Holiday	**Mar 25 - Holiday**	**Mar 27 - Holiday**
Mar 30 Project check ins. **Weekly check ins + milestone reports begin**	**Apr 01** Project check ins.	**Apr 03**
Apr 06 Project check ins. **Milestone 1 due**	**Apr 08** Project check ins.	**Apr 10**
Apr 13 Project check ins. **Milestone 2 due**	**Apr 15** Project check ins.	**Apr 17** **Milestone 3 proposal due**
Apr 20 - Holiday	**Apr 22** Project check ins (optional?).	**Apr 24**
Apr 27 Project check ins. **Milestone 3 due**	**Apr 29** Project check ins.	**May 01** **Project report due**
May 04 Final project presentations.	**May 06** Final project presentations.	**May 08**
May 11 Final project presentations. **LDOC!**		

Project Deliverables

- May 1 – Project Report
  - 4 pages, not including references
    - Grad teams get 1 additional page for overview of related works
  - References can take up additional page(s) [unlimited]
  - Use template (will be provided)
  - Technical summary sheet (does not count towards page limit)

- May 4, 6 & 11 – Poster Presentations

- May 11 – Peer Evaluation Form & Review of Other Projects

Report/Poster Requirements

- Clearly introduce the problem and why it is important
- Describe and cite previous work– how is it different from what you did in the project
- Describe the three most interesting insights / innovative ideas pursued in your project
  - Make sure you indicate what parts worked and what parts did not work
- Provide any suggestions for future work
- Make the figures clear – i.e., keep the font large enough for readability, do not rely on color for your figures, do not try to include too much detail

Technical Summary Sheet (1-page)

- Project Title
- Authors
- List the three key insights/innovations of project
- List the contributions of each partner
- Provide name of project repository submission

Grading

- **(30%) Progress**
  - (15%/10%/5%) for each check in completion (attendance, **discussion**, progress)
    - Half mark: write a milestone report
    - Half mark: discussion about report (including **addressing questions**)
- **(65%) Quality of the project** → Note: A large portion of grade is based on ability to answer questions during poster session on the categories below [ALL TEAM MEMBERS MUST ATTEND POSTER SESSION (missing member will receive zero)]
  - Report + poster should cover the following (template with be provided)
    - (10%) Background and motivation: related work (grad-1 page) and contributions
    - (10%) Description of experiment setup. Are the experiments unbiased? Any assumptions made? Did you use reasonable baselines?
    - (15%) Quantified results across relevant metrics (e.g., accuracy, energy, latency, area). Demonstrate a set of results that back your contributions. Describe the trade-off.
    - (30%) Comprehensive exploration, analysis and **interpretation of results. Detailed insights.** Where would this approach be beneficial? Where would it not be beneficial?
- **(5%) Community Support**
  - Runnable, well documented, reproducible (get the same figures), and students next semester can use it as an example.
  - If use LLM, provide requested info (see additional slide provided by TAs)
  - Review of other projects
- ***Note: Grade maybe adjusted based on peer evaluation***

Published Design Projects!

Architecture-Level Modeling of Photonic Deep Neural Network Accelerators

Tanner Andrulis, *MIT*, Cambridge, USA, andrulis@mit.edu
Gohar Irfan Chaudhry, *MIT*, Cambridge, USA, girfan@mit.edu
Vinith M. Suriyakumar, *MIT*, Cambridge, USA, vinithms@mit.edu
Joel S. Emer, *MIT, Nvidia*, Cambridge, USA, jsemer@mit.edu
Vivienne Sze, *MIT*, Cambridge, USA, sze@mit.edu

***Abstract*—Photonics is a promising technology to accelerate Deep Neural Networks as it can use optical interconnects to reduce data movement energy and it enables low-energy, high-throughput optical-analog computations.**

To realize these benefits in a full system (accelerator + DRAM), designers must ensure that the benefits of using the electrical, optical, analog, and digital domains exceed the costs of converting data between domains. Designers must also consider system-level energy costs such as data fetch from DRAM. Converting data and accessing DRAM can consume significant energy, so to evaluate and explore the photonic system space, there is a need for a tool that can model these full-system considerations.

In this work, we show that similarities between Compute-in-Memory (CiM) and photonics let us use CiM system modeling tools to accurately model photonics systems. Bringing modeling tools to photonics enables evaluation of photonic research in a full-system context, rapid design space exploration, co-design, and comparison between systems.

Using our open-source model, we show that cross-domain conversion and DRAM can consume a significant portion of photonic system energy. We then demonstrate optimizations that reduce conversions and DRAM accesses to improve photonic system energy efficiency by up to 3×.

***Index Terms*—photonics, optical computing, photonic computing, compute-in-memory, modeling, accelerator**

I. Introduction

Deep Neural Networks (DNNs) can be energy-intensive to compute due to the movement of large tensors and the many multiply-accumulate (MAC) operations that they require. To address these challenges, photonic systems (accelerator + DRAM) leverage the digital-electrical (*DE*), analog-electrical (*AE*), digital-optical (*DO*), analog-optical (*AO*) domains. Specifically, optical (*i.e.*, *DO* and *AO*) interconnects can

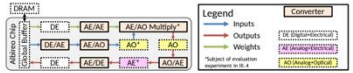

Fig. 1. Albireo architecture. As data traverse the *DE*, *AO*, and *AE* domains, they leverage different movement and reuse opportunities but pay energy for data converters, notated *X*/*Y* for conversion from domain *X* to domain *Y*.

optical resonators), architecture (*i.e.*, what components are used, how many components, how they connect), workload (*i.e.*, DNN layer types, tensor shapes/values), and mapping (*i.e.*, how the workload is scheduled onto the architecture).

Fortunately, these characteristics are not unique to photonics. Analog Compute-in-Memory (CiM) systems have a large full-system co-design space, leverage the advantages of multiple domains (*AE* and *DE*), and face the challenge of high cross-domain conversion energy.

In this work, we show that these similarities let us leverage the open-source CiMLoop [1]–[4] tool to accurately model photonic systems. Bringing this tool to photonics enables researchers to **(1)** accurately evaluate and compare research contributions in a full-system context (*e.g.*, see how a novel component affects a full system or compare two photonic systems across a range of DNN workloads) **(2)** perform fast design-space exploration over the large co-design space [1], and **(3)** share knowledge between the photonics and CiM research communities.

II. Photonics Modeling Tool

The tool takes as input specifications of a DNN workload, components, and architecture as defined in Section I. The tool

ISPASS 2024

Ultra-low Power Superconducting Electronics for Deep Learning Accelerator Architectures: Evaluating Energy Efficiency and Scalability

9.22

L. Camron Blackburn, Evan Golden, Tanner Andrulis, Vivienne Sze, Joel Emer, Neil Gershenfeld, Karl K. Berggren

Sponsorship: MIT Lincoln Laboratory, the MIT AI Hardware Program

Since the invention of the Josephson junction in the 1960s, superconducting electronics have shown promise for high-speed and energy-efficient computing. Since 2013, the Adiabatic Quantum Flux Parametron (AQFP) device has gained popularity for its ultra-low energy dissipation. AQFP inverters dissipate 10^{-21}J per switching event, 100× less than other superconductor logic, and 10^{6}× less energy than modern-day CMOS transistors or 10^{3}× when including the cryogenic cooling cost. As Moore's law ends and energy efficiency emerges as a limit on today's computing systems, superconducting AQFP logic is a promising technology to address these energy challenges. Although individual AQFP device performance is impressive, superconducting electronics have failed to replace CMOS systems in the past in part due to the high cost of cryogenic low-noise testing environments and the limitations of superconductor memory scaling. To realize the promise of superconducting electronics, there is a need to architect full systems that can leverage the benefits of the unique superconductor physics (e.g., low-energy logic, low-energy interconnects on zero-resistance wires) while addressing the challenges (e.g., using low-noise cryogenic environments commoditized by the quantum computing industry, constructing a memory hierarchy that addresses the lack of a scalable, high-density superconducting memory). In this work, we extend Timeloop/Accelergy accelerator modeling tools to support superconducting accelerators. This framework explores the design space of deep learning accelerator architectures with a toolbox of superconducting circuits from various logic families. We present results demonstrating the tradeoffs between superconductor vs. CMOS accelerators while running a range of deep learning workloads.

MARC 2025