| Project Name | Brief Project Description | Primary Advisor |
|---|---|---|
| **RAG on the EDGE** | Implement Retrieval-Augmented Generation (RAG) on AMD NPU, Qualcomm QIDK, and NVIDIA Jetson platforms, followed by a comparative performance evaluation across these edge accelerators. | Suresh Purini |
| **RAG Accelerator on Xilinx Alveo U50** | Implement a hardware accelerator for Retrieval-Augmented Generation (RAG) on the Xilinx Alveo U50 FPGA, optimizing compute and memory usage to improve latency, throughput, and energy efficiency. Compare its performance against GPU- and CPU-based implementations using available frameworks, without requiring custom accelerator development on those platforms. | Suresh Purini |
| **RAG Accelerator on Xilinx Versal VCK5000** | Develop a Retrieval-Augmented Generation (RAG) accelerator on the Xilinx Versal VCK5000, leveraging AI Engine (AIE) tiles for high-performance matrix operations and programmable logic for data movement. Benchmark its latency, throughput, and energy efficiency against GPU- and CPU-based implementations using existing frameworks, without building them from scratch. | Suresh Purini |
| **BitNet 1.58 on Xilinx Alveo U50** | Implement the BitNet 1.58 large language model with ternary weight quantization on the Xilinx Alveo U50 FPGA. Focus on optimizing compute and memory efficiency to enable scalable inference. Performance will be evaluated in terms of latency, throughput, and energy efficiency, with comparisons to CPU and GPU baselines. Refer the latest paper **TerEffic: Highly Efficient Ternary LLM Inference on FPGA.** | Suresh Purini |
| **BitNet 1.58 on Xilinx Versal VCK5000** | Deploy the BitNet 1.58 large language model with ternary weight quantization on the Xilinx Versal VCK5000, leveraging AI Engine (AIE) tiles for high-performance matrix operations and programmable logic for efficient dataflow. The design will be benchmarked for latency, throughput, and energy efficiency, and compared against CPU and GPU implementations. Refer the latest paper **TerEffic: Highly Efficient Ternary LLM Inference on FPGA.** | Suresh Purini |
| **BitNet 1.58 on AMD NPU** | Implement the BitNet 1.58 large language model with ternary weight quantization on the AMD NPU platform, leveraging its specialized matrix engines for efficient inference. Optimize for latency, throughput, and energy efficiency, and compare against FPGA (U50, VCK5000), GPU, and CPU implementations. Refer the latest paper **TerEffic: Highly Efficient Ternary LLM Inference on FPGA.** | Suresh Purini |

| Project Name | Brief Project Description | Primary Advisor |
|---|---|---|
| **Large Language Processor (LLP) - all digital** | Develop a streamlined **all-digital** accelerator hardware, custom instruction and dataflow for high-speed, **long-context,** mixture-of-experts LLM inference.<br>Platform(s): gem5, firesim, hdl/System C<br>**Ref: LPU: A Latency-Optimized and Highly Scalable Processor for Large Language Model Inference** | Priyesh Shukla |
| **Large Language Processor (LLP) - IMC** | Develop a streamlined **in-memory computing**-based accelerator hardware, custom instruction set and LLM-specific dataflow for high-speed, **long-context** LLM inference.<br>Platforms: gem5, hspice/cadence, hdl/system C<br>**Ref: LPU: A Latency-Optimized and Highly Scalable Processor for Large Language Model Inference** | Priyesh Shukla |
| **Vision AI accelerator for AR/VR** | Develop a generic DNN hardware prototype to accelerate low-power computations in 3D vision pipeline for AR/VR glasses.<br>**Refer to CICC2022 paper from Meta Reality Labs: System-Level Design and Integration of a Prototype AR/VR Hardware Featuring a Custom Low-Power DNN Accelerator Chip in 7nm Technology for Codec Avatars** | Priyesh Shukla |
| **Visual SLAM AI accelerator for Micro-robotics** | ASIC/FPGA/EdgeAI/RISC-V implementation of deep learning based SLAM algorithm for low-power, accurate and on-device navigation of tiny drones.<br>**Ref: NanoSLAM: Enabling Fully Onboard SLAM for Tiny Robots** | Priyesh Shukla |
| **On-device training of DNNs** | Design accelerator for backpropagation in Convolutional and fully-connected layers to carry out incremental training locally at the edge<br>**Ref: DARKSIDE: A Heterogeneous RISC-V Compute Cluster for Extreme-Edge On-Chip DNN Inference and Training**<br><br>**Mandheling: Mixed-Precision On-Device DNN Training with DSP Offloading** | Priyesh Shukla |
| **Deep learning operator library and RISC-V custom extensions** | Design multi-specification (low power, high speed, variying precision…) library of deep learning operators - 2D/3D convolution (point-wise, depthwise separable…), Self-attention, Multi-head attention for Transformers | Priyesh Shukla |

| Project Name | Brief Project Description | Primary Advisor |
|---|---|---|
| **KV Cache Optimization** | Investigate various algorithmic/deep learning architecture (CNNs/LLMs) optimization schemes for efficient processing at the edge<br><br>**Refs:**<br>**DEEP COMPRESSION: COMPRESSING DEEP NEURALNETWORKS WITH PRUNING, TRAINED QUANTIZATIONAND HUFFMAN CODING**<br><br>**An Efficient Hybrid Deep Learning Accelerator for Compactand Heterogeneous CNNs**<br><br>**DeepSeek-V3 Technical Report**<br><br>**EIE: Efficient Inference Engine on Compressed Deep Neural Network**<br><br>**Integer quantization for deep learning inference: Principles and empirical evaluation** | Girish Varma |
| **Neural Architecture Search** | | |
| **Quantization aware training and inference** | | |
| **Pruning and Compression Schemes** | | |

| | | |
|---|---|---|
| **Design Space Exploration of GEMM on Alveo U50 (FP32 & INT8)** | Implement and optimize GEMM kernels on the Xilinx/AMD Alveo U50, exploring FP32 (via floating-point IP) and INT8 (INT32 accumulation) variants. | Suresh Purini |
| **Design Space Exploration of GEMM on Versal VCK5000 (FP32 & INT8)** | Implement and optimize GEMM kernels on the Xilinx Versal VCK5000, exploring FP32 and INT8 variants. The project leverages AI Engine (AIE) tiles for high-throughput matrix operations and the programmable logic (PL) for memory orchestration and data movement. | Suresh Purini |
| **Sparse Matrix–Dense Matrix Multiplication (SpMM) Accelerator on Alveo U50** | Design and implement a hardware accelerator for Sparse Matrix–Dense Matrix Multiplication (SpMM) on the Xilinx Alveo U50 FPGA. Explore different sparse storage formats (CSR, CSC, Blocked-CSR) and scheduling strategies to optimize compute efficiency and memory bandwidth utilization. Evaluate performance in terms of latency, throughput, and energy efficiency, and compare against optimized CPU (MKL) and GPU (cuSPARSE) implementations. Refer **FPGA-Based Sparse Matrix Multiplication Accelerators: From State-of-the-Art to Future Opportunities** | Suresh Purini |
| **Sparse Matrix–Dense Matrix Multiplication (SpMM) Accelerator on Versal VCK5000** | Develop a SpMM accelerator on the Xilinx Versal VCK5000, leveraging AI Engine (AIE) tiles for dense matrix computation and programmable logic for sparse indexing and dataflow control. Investigate design choices such as tiling, load balancing across sparse rows, and hybrid AIE–PL partitioning. Benchmark latency, throughput, and energy efficiency, and compare against CPU and GPU baselines to highlight the benefits of Versal's heterogeneous architecture. Refer **FPGA-Based Sparse Matrix Multiplication Accelerators: From State-of-the-Art to Future Opportunities** | Suresh Purini |
| **Graph Neural Network (GNN) Accelerator on Alveo U50** | Design and implement a GNN inference accelerator on the Xilinx Alveo U50 FPGA. The focus is on optimizing the sparse-dense matrix multiplication and neighbor aggregation kernels that dominate GNN workloads. Explore different sparse formats (CSR, COO) and tiling strategies to balance memory bandwidth and compute utilization. Evaluate the accelerator on standard GNN models (e.g., GCN, GraphSAGE) and benchmark performance against CPU and GPU baselines. Refer **A Review of FPGA-based Graph Neural Network Accelerator Architectures** | Suresh Purini |

| | | |
|---|---|---|
| **Chipyard** | Develop RISC-V Co-processor (GEMM accelerator) in Chipyard to accelerate DNN training and inference<br>**Ref: https://chipyard.readthedocs.io/en/stable/index.html** | Priyesh Shukla |
| **Chipyard** | Develop In-memory computing macro as RISC-V co-processor in Chipyard for DNN training and inference<br>**Ref: https://chipyard.readthedocs.io/en/stable/index.html** | Priyesh Shukla |
| **In-memory computing** | ADC-free IMC macro for DNN training and inference capable of performing analog computations and digitization by exploiting memory bank characteristics<br>**Ref: HCiM: ADC-Less Hybrid Analog-Digital Compute in Memory Accelerator for Deep Learning Workloads**<br><br>**A Brain-Inspired ADC-Free SRAM-Based In-Memory Computing Macro With High-Precision MAC for AI Application** | Priyesh Shukla |
| **High-bandwidth processing-in-memory (HBM-PIM) simulation and characterization** | Gem5 simualtion framework for HBM-PIM simulation and characterization for deep learning inference<br>**Ref: https://www.gem5.org/assets/files/hpca2023-tutorial/gem5-tutorial-hpca-2023.pdf**<br><br>**Gem5-X: A Many-core Heterogeneous Simulation Platform for Architectural Exploration and Optimization**<br><br>**https://www.gem5.org/documentation/general_docs/stdlib_api/gem5.components.memory.hbm.html** | Priyesh Shukla |
| **Bitnet (1-bit LLM) orchestration on multi-core RISC-V accelerator-vector cores** | Enhance PULP's ARA multicore RISC-V architecture with custom accelerator extension and implement 1-bit LLM on the RISC-V architecture<br>**Ref: Ara2: Exploring Single- and Multi-Core Vector Processing With an Efficient RVV 1.0 Compliant Open-Source Processor** | Priyesh Shukla |

| | | |
|---|---|---|
| **Graph Neural Network (GNN) Accelerator on Versal VCK5000** | Develop a GNN inference accelerator on the Xilinx Versal VCK5000, leveraging AI Engine (AIE) tiles for dense feature transformations and programmable logic for sparse neighbor aggregation. Investigate partitioning strategies across AIE and PL, dataflow optimizations, and load balancing for irregular graph structures. Benchmark throughput, latency, and energy efficiency using real graph datasets, and compare against state-of-the-art CPU and GPU implementations. Refer A Review of FPGA-based Graph Neural Network Accelerator Architectures | Suresh Purini |
| **SCALE: Systolic Compiler + Array co-LEarning for CNNs on FPGAs** | Develop a parametric systolic-array (SA) accelerator and a companion mapping compiler for CNN inference. Explore different dataflows (output-, weight-, row-stationary), array shapes, on-chip tiling, and quantization (INT8/INT4). The compiler should generate SA schedules via loop transformations and DMA overlap. Evaluate on U50 and/or VCK5000 with models like ResNet-18 and MobileNet-V2, reporting latency, throughput, utilization, and energy per inference. Refer **Automated Systolic Array Architecture Synthesis for High Throughput CNN Inference on FPGAs** | |

| | | |
|---|---|---|
| **Brain-inspired (spiking) neural network on FPGA/IMC** | Implement neuromorphic computing accelerator on Xilinx Alveo exploiting spike time-dependent elasticity with higher energy efficiency<br>**Refs: DeepFire2: A Convolutional Spiking Neural Network Accelerator on FPGAs**<br><br>**IzhiRISC-V - a RISC-V-based Processor with Custom ISA Extension for Spiking Neuron Networks Processing with Izhikevich Neurons** | Priyesh Shukla |
| **Photonic neural accelerator** | Implement a simulation framework for photonic computations to accelerate CNNs<br>**Ref: LightML: A Photonic Accelerator for Efficient General Purpose Machine Learning** | Priyesh Shukla |
| **NVM (non-volatile memory) crossbars** | Explore NVM technology (PCM, ReRAM, MRAM...) to for analog in-memory computing to accelerate CNNs and transformers<br>**Ref: Memory devices and applications for in-memory computing** | Priyesh Shukla |
| **Chiplets - Wafer-scale computing** | Explore chiplets - heterogenous integration, partitioning, scheduling, network routing strategies for efficient LLM/VLM inference<br>**Ref: WSC-LLM: Efficient LLM Service and Architecture Co-exploration for Wafer-scale Chips** | Priyesh Shukla |
| **KV Cache management for LLMs** | Explore strategies to reduce redundant computations and improve memory utilization during long-context real-time LLM inferencing<br>**Ref: A Survey on Large Language Model Acceleration based on KV Cache Management** | Priyesh Shukla |
| **Toolchain optimization passes for CNNs and LLMs** | Compilers (MLIR, IREE, LLVM, GCC)/OS - optimizations for new accelerator co-processor<br>**Refs: An MLIR-based Compiler Flow for System-Level Design andHardware Acceleration**<br><br>**https://github.com/iree-org/iree** | Priyesh Shukla |