|  |  |
| --- | --- |
| A close up of a sign  Description automatically generated1859 Shirley Lane,11-B8, Ann Arbor, MI, 48105kanzhu@umich.edu(734) 596-2015OBJECTIVEA self-motivated undergraduate student applying for the architecture, operating systems, compilers, and hardware security Ph.D. program. INTERESTS **-** **Micro-architecture**- Prefetchers- Accelerators- Branch Predictors**- Operating systems****- Compilers****- Datacenter****- Hardware security**SKILLS COMPUTERC, C++, Verilog, Assembly, Java, Python, MATLAB, Embedded system, Git, Latex, ScriptsLANGUAGESEnglish – Full Professional Proficiency Mandarin – Native fluency **RELEVANT COURSES**Computer Architecture, Intro to Operating Systems, Formal Verification, Compiler Construction, Embedded System Design, Analog Circuits, Instruction to Signal and Systems, Quantum Mechanics  |  **Kan Zhu****EDUCATION****University of Michigan – GPA 4.00/4.00 Ann Arbor, MI**Major: BS Computer Engineering **September 2021 – Sept 2023****Shanghai Jiao Tong University – GPA 3.82 / 4.00 Shanghai, China**Major: BS Electrical and Computer Engineering  **September 2019 – September 2021**Rank: 10 / 300**AWARDS****ACM Student Research Competition 1st Place Award Chicago, IL****MICRO 2022, Undergraduate Division October 2022****Present a poster and give 10 min talk on micro-architectural implications of Google applications** **PUBLICATIONS****Google Data Center Applications Analysis** **In Submission to ASPLOS 2023(Name omitted to maintain anonymity) October 2022****Lead author: Kan Zhu****Micro-op Cache Replacement Policy** **In Submission to ISCA 2023 (Name omitted to maintain anonymity) November 2022****Second author: Kan Zhu****RESEARCH EXPERIENCE****EFESLAB, University of Michigan Ann Arbor, MI****Advisors: Tanvir Ahmed Khan and Baris Kasikci****Colaberators: Heiner Litz (UCSC), Shuwen Deng (Tsinghua Unversity), Akshitha Sriraman (CMU), Derek Bruening (Google), Victor Lee(Google)****Optimizing the performance of Google web services May 2022 – October 2022*** Led the project, conducting experiments and analysis.
* Investigated the thread switch behavior and its performance implication
* Identified the performance bottleneck and optimization directions
* Evaluated state-of-the-art prefetchers and replacement policies
* Classified workload and create a representative workload subset

**Optimizing micro-op cache for data center applications October 2022 – November 2022*** Understand the limitations of state-of-the-art replacement policy and the uniqueness of micro-op cache.
* Proposed and evaluated counter-based, profile-guided replacement policy

**PROFESIONAL EXPERIENCE** **Shanghai Jiao Tong University Shanghai, China** **VG101 Introduction to Programming teaching assistant May 2021 – July 2021*** Composed lab materials and designed class exercises
* Lead coding lab sections to help students practice programming skills
* Held recitation classes to summarize the key class points for 30 students every week

**RELEVANT COURSE PROJECTS**  **EECS 470 – COMPUTER ARCHITECTURE (A+) Ann Arbor, MI****R10K Based Out of Order Processor January 2022*** Implemented a ROB, RS, Map Table, Arch. Map Table and Free List, LSQ, D-Cache, I-Cache, together with the necessary Functional Units to deal with high memory latency
* Included features such as a 2-way superscalar to support multiple instructions, early branch resolution using b-mask to avoid flush/ squash delays, non-blocking I-cache and D-cache, pipeline memory access, and a GUI debugger to improve efficiency

**EECS 482 – Introduction to Operating Systems (A+) Ann Arbor, MI****Operating System Components January 2022** * Practiced multithreading programming
* Developed a thread library to provide mutex and conditional variable interfaces
* Implemented a pager to manage memory space for multiple processes
* Created a network file server handling concurrent user requests

**EECS 483 – Compiler Construction (A+) Ann Arbor, MI****Decaf Compiler January 2022*** Created a Lexical analyzer using Flex and Syntax analyzer using Bison
* Implemented a Semantic analyzer, TAC code generator, and Code optimizer
* Included features such as class inheritance and polymorphism
 |