David Kebo Houngninou, Ph.D.

Instructional Associate Professor
Department of Computer Science and Engineering · Texas A&M University · College Station, Texas
davidkebo@tamu.edu · 214-686-9611 · ORCID 0000-0002-7017-8440
people.engr.tamu.edu/davidkebo · davidkebo.com

EDUCATION

Ph.D., Computer Engineering — Southern Methodist University, Dallas, Texas, December 2017
Area of research: Hardware Formal Verification

M.S., Computer Engineering — Washington University in St. Louis, Missouri, December 2010

B.S., Computer Engineering — University of Evansville, Indiana, December 2008

Licence, Electronics — Université de Bourgogne, France, 2005

ACADEMIC APPOINTMENTS

Instructional Associate Professor

Department of Computer Science and Engineering, Texas A&M University · August 2018 – Present

Teach core undergraduate and graduate courses in hardware design verification, computer organization and architecture, operating systems, data structures, algorithms, and computer security. Develop curriculum, laboratory assignments, and assessments that build both theoretical knowledge and practical skill.

- Designed the department's graduate hardware verification sequence from scratch, including labs, assessments, and supporting tooling.
- Co-developed the graduate emulation and prototyping course with Synopsys, built on the ZeBu emulation platform.
- Embedded official Cadence SystemVerilog and UVM certification tracks into CSCE 616 and CSCE 714, with laboratories on Xcelium and vManager.
- Founded and direct WAVECHIP, a train-the-trainer verification workforce program serving degree students, working professionals, and faculty across four-year colleges, community colleges, and industry workforce programs.
- Lead a research group publishing on machine-learning-augmented verification: multi-agent LLM pipelines for RTL bug synthesis, coverage closure, and specification-to-assertion translation.
- Advise graduate students and serve as first-line technical support for verification flows across every cohort.

Adjunct Lecturer

Department of Computer Science and Engineering, Southern Methodist University · August 2015 – May 2018

Developed and taught undergraduate and graduate courses in microprocessor architecture and interfacing, digital systems design, software engineering, and discrete computational structures. Prepared curriculum covering ARM architecture, FPGA architecture, and combinatorics.

INDUSTRY EXPERIENCE

Design Engineer, Multicore and DSP Design and Verification

Texas Instruments, Dallas, Texas · May – September 2017

Verified multicore SoC and DSP architectures including Cortex-R5 processor cores and PCIe interfaces. Developed directed and constrained-random test stimulus to exercise cache coherency protocols and

performed pre-silicon verification to identify complex multi-core interaction defects before tape-out.

Researcher, Temporal and Structural Graph Analysis
IBM Research Lab, Cambridge, Massachusetts · June – September 2014

Built interactive D3.js visualizations of governance, risk, and compliance data, enabling dynamic sorting and analysis of risks, ratings, and comments with exportable tables for non-technical decision-makers.

Software Developer
NXP Semiconductors, Austin, Texas · May – August 2012

Developed and maintained Grails-based applications implementing MVC frameworks with Groovy and Java, integrating RESTful APIs for the NXP semiconductor fabrication plant.

Software Developer
Wirevibe, Irving, Texas · January – October 2011

Maintained back-end systems for web applications using PHP, MySQL, RESTful APIs, and MVC architectures.

Hardware Engineer
Magellan Integration, Evansville, Indiana · August 2008 – May 2010

Led IT infrastructure and lab design projects covering networking, security, and surveillance systems. Directed design and documentation for CCTV low-voltage installations across multiple corporate locations.

PUBLICATIONS

Peer-Reviewed Conference Papers

Houngninou, D. K., Jasper, S., Luu, M., Pan, E., Tyagi, A., Quinn, M., & Hu, J. (2025). BugGen: A self-correcting multi-agent LLM pipeline for realistic RTL bug synthesis. *Proceedings of the 7th ACM/IEEE International Symposium on Machine Learning for CAD (MLCAD 2025).*

Houngninou, D. K., Liu, C., Parlapalli, J. P., Quinn, M., Tyagi, A., & Hu, J. (2025). Improving last-mile coverage in functional verification. *Proceedings of the 7th ACM/IEEE International Symposium on Machine Learning for CAD (MLCAD 2025).*

Houngninou, D. K. (2023). FLIP: A RISC-V visual computer architecture simulator for K-12. *Proceedings of the 54th ACM Technical Symposium on Computer Science Education, V.2 (SIGCSE 2023),* 1271. https://doi.org/10.1145/3545947.3573252

Houngninou, D. K., Holanda, M., & Da Silva, D. (2023). Early introduction to computer architecture in K-12. *Proceedings of the 54th ACM Technical Symposium on Computer Science Education, V.2 (SIGCSE 2023),* 1331. https://doi.org/10.1145/3545947.3576277

Houngninou, D. K., & Thornton, M. A. (2017). Simulation of switching circuits using transfer functions. *2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS),* Boston, MA, 511–514.

Houngninou, D. K., & Thornton, M. A. (2016). Implementation of switching circuit models as transfer functions. *2016 IEEE International Symposium on Circuits and Systems (ISCAS),* Montreal, QC, 2162–2165.

Book Chapters

Thornton, M. A., Houngninou, D. K., & Miller, D. M. (2022). Computing the Reed-Muller spectrum / algebraic normal form: Functional methods. In B. Steinbach (Ed.), *Advances in the Boolean Domain.* Cambridge Scholars Publishing.

Houngninou, D. K., Thornton, M. A., & Miller, D. M. (2022). Extracting the Reed-Muller spectrum / algebraic normal form from a circuit specification. In B. Steinbach (Ed.), *Advances in the Boolean*

Domain. Cambridge Scholars Publishing.

Journal Articles

Houngninou, D. K., Miller, D. M., & Thornton, M. A. (2021). ANF computation of cryptographic switching functions using a netlist representation. *Bell System Technical Journal, 28*(1), 59–98.

Preprints

Spec2Assertion: Automated translation of natural-language specifications into formal assertions. (2025).

INVITED TALKS AND CONFERENCE WORKSHOPS

Early Introduction to Computer Architecture in K-12
SIGCSE Technical Symposium on Computer Science Education · March 2023

An educational framework for K-12 and undergraduate students to learn computer architecture by building custom processors and observing programs simulated in real time.

FLIP: A RISC-V Visual Computer Architecture Simulator for K-12
SIGCSE Technical Symposium on Computer Science Education · March 2023

An interactive simulator allowing students to build and run programs on a custom RISC-V processor.

A RISC-V Open-Source Architecture Design Space Exploration Toolbox
CMD-IT/ACM Richard Tapia Celebration of Diversity in Computing · September 2021

Hands-on workshop introducing Trireme, a RISC-V architecture design exploration suite for education and research.

EDUCATIONAL PLATFORMS AND PROGRAMS

WAVECHIP — Hardware Verification Workforce Training
wavechip.tamu.edu

Verification curriculum for semiconductor workforce development, built on a train-the-trainer model. Introduces hardware verification courses to undergraduate programs and community colleges that traditionally do not offer them, and prepares faculty with teaching materials and ongoing support. Covers functional verification, emulation-based verification, and the integration of AI and machine learning in verification.

Formal — Browser-Based Formal Verification Environment
formal.org

Curriculum and hands-on platform teaching learners to write SystemVerilog assertions, run SymbiYosys, and diagnose design bugs through browser-based labs, removing the toolchain setup and license cost that normally gate formal methods.

OpenDV — Open Verification Education Platform
opendv.net

Open-access platform making structured hardware verification education available to engineers outside a university enrollment.

FLIP — RISC-V Visual Computer Architecture Simulator
Web-based educational simulator with independently usable teaching tools. A chip builder and navigator let learners assemble a custom system from ALU, caches, memory, and configuration parameters, then inspect the live state of RAM, registers, cache, and buses during execution. A built-in code editor and simulator visualize assembly execution step by step with controllable speed and detail.

Horus

horus.teamup.org

Public tool built by the Teamup 2026 cohort on the Anthropic Economic Index. Reads a college degree course by course and shows which of the skills it teaches AI is already changing: 285 majors, from 11,320 course descriptions matched against the Anthropic Economic Index and the U.S. Department of Labor O*NET catalog.

COURSE DEVELOPMENT

CSCE 689 — Hardware-Based Verification Using Emulation and Prototyping
Texas A&M University · August 2023

Graduate course developed in collaboration with Synopsys teaching state-of-the-art FPGA-based emulation systems for verification, aligned with both academic theory and current industry standards.

CSCE 110 — Programming I, online course
Texas A&M University · June 2020

Online introductory Python course for students with little or no programming experience. Developed recorded lectures, lecture notes, auto-graded labs, and practice activities.

COURSES TAUGHT

Texas A&M University

CSCE 714 — Advanced Hardware Design Functional Verification. Industry-standard verification using Cadence Xcelium and vManager with UVM and SystemVerilog. Official Cadence certifications; laboratories verifying AMBA protocols, multicore cache systems, and Verification IP integration.

CSCE 689 — Hardware-Based Verification Using Emulation and Prototyping. FPGA-based emulation for IP and SoC verification and system validation on the ZeBu platform.

CSCE 616 — Introduction to Hardware Design Verification. Functional verification fundamentals; testbenches for real protocols (HTAX), assertion-based verification, coverage analysis, constrained randomization, UVM sequences, and regression.

CSCE 465 — Computer and Network Security. Operating system and program security, malware, cryptography, authentication, firewalls, intrusion detection, and application security.

CSCE 410 — Operating Systems. System calls, processes, memory management, scheduling, threading, synchronization, file systems, and I/O. Laboratories extend open-source kernels such as RISC-V xV6.

CSCE 313 — Introduction to Computer Systems. Operating system application interface, inter-process communication, systems and network programming, and introductory security concepts.

CSCE 312 — Computer Organization. Data representation, machine language, processor architecture, memory hierarchy, assemblers, virtual machines, and compiler design.

CSCE 222 — Discrete Structures for Computing. Logic, proofs, sets, algebraic structures, graph theory, combinatorics, finite state machines, and Turing machines.

CSCE 221 — Data Structures and Algorithms. Abstract data types, sorting and selection, searching, graphs, hashing, and asymptotic analysis in C++.

CSCE 206 — Structured Programming in C. Internal data representation, software design principles, and structured and object-oriented programming in C.

CSCE 121 — Introduction to Programming Design and Concepts. Algorithm design and implementation, program control, functions, classes, exceptions, abstraction, modularity, and debugging.

CSCE 110 — Programming I. Computational thinking and Python programming for students with little or no prior experience.

Southern Methodist University

CSE/EE 7385 — Microprocessor Architecture and Interfacing. ARM architecture, memory structure and interfacing, bus systems, and a laboratory designing interfaces to processors, memories, and peripherals.

CSE/EE 7387 — Digital Systems Design. Combinational logic synthesis in Verilog, FPGA architecture, and finite-state machine design, with laboratory experiments and a final design project.

CSE 4345 — Software Engineering. Development models, system feasibility and requirements, architecture and design, validation and verification, maintenance, and project management.

CSE 2353 — Discrete Computational Structures. Sets, logic, proofs, graph theory, and combinatorics applied to computer science problems.

SERVICE AND PROFESSIONAL ACTIVITIES

Founder and President

Teamup, a 501(c)(3) nonprofit, College Station, Texas · January 2022 – Present

Lead a nonprofit promoting project-based learning and preparing K-12 and college students for STEM careers while empowering them to build technology for social good.

- Designed and run the production process behind the Apps for Good program: two-week cycles, daily check-ins, weekly written progress reports, and mandatory peer review before release. Seven cohorts, 176 student builders, 15 published applications.
- Set the AI-in-the-loop standard on a shipped product: AI drafts, a student verifies every result by hand, and no model runs at request time, so every published figure is traceable to its source.
- Secured and administered a Greater Texas Foundation grant with a full reported outcome; program an industry guest-speaker series and publish a written retrospective each cycle.
- Run international cohorts with Texas A&M and EPITECH Bénin students, each mentoring a high school participant one-to-one on the same live product.

Doctoral Consortium Chair and Deputy Program Chair

CMD-IT/ACM Richard Tapia Celebration of Diversity in Computing Conference · January 2023 – Present

Lead the doctoral consortium, where Ph.D. candidates present research proposals and results to a panel of researchers and industry professionals. As Deputy Program Chair, directed the technical program committee: set the review bar, recruited and coordinated volunteer reviewers, and delivered the program to a fixed conference date.

Mentor

The Posse Foundation · March 2024 – Present

Provide academic advising to Posse scholars throughout their studies at Texas A&M through weekly group sessions and individual meetings, connecting scholars to campus resources.

Diversity, Equity, and Inclusion Committee

Department of Computer Science and Engineering, Texas A&M University · August 2022 – Present

Advise department leadership on recruiting and retaining underrepresented groups, equity in resource allocation, and inclusive departmental culture; track representation data and develop evidence-based recommendations.

Undergraduate Curriculum and ABET Committee

Department of Computer Science and Engineering, Texas A&M University · August 2021 – Present

Review proposed updates to the computer science undergraduate curriculum and coordinate with the Accreditation Board for Engineering and Technology.

Undergraduate Admissions Committee

Department of Computer Science and Engineering, Texas A&M University · August 2021 – Present

Review candidates for admission into the computer science and engineering program.

Computer Engineering Coordinating Committee
Department of Computer Science and Engineering, Texas A&M University · August 2018 – May 2019
Reviewed proposed changes to the computer engineering undergraduate curriculum and coordinated with the Accreditation Board for Engineering and Technology.

TECHNICAL EXPERTISE

Simulation and debug. Cadence Xcelium and vManager; Synopsys VCS and Verdi; Siemens EDA Questa.

Formal verification. Cadence JasperGold; Synopsys VC Formal; SymbiYosys; SystemVerilog Assertions, property specification, and proof debug.

Emulation and prototyping. Synopsys ZeBu, FPGA-based emulation, co-emulation, design and testbench preparation, compile and runtime flows.

Methodologies. UVM, constrained-random verification, assertion-based verification, coverage-driven verification and closure, verification planning, regression management.

Protocols and designs. AMBA/AXI, PCIe, HTAX, multicore cache coherency, Cortex-R5, DSP subsystems.

Languages. SystemVerilog, Verilog, VHDL, SystemC, C/C++, Python, Tcl, JavaScript.

PROFESSIONAL MEMBERSHIPS AND LANGUAGES

American Society for Engineering Education (ASEE)

ACM Special Interest Group on Computer Science Education (SIGCSE)

English and French — full professional proficiency