2026 ASEE ANNUAL CONFERENCE June 21–24, 2026 | Charlotte Convention Center, NC
Host Sponsor UNC Charlotte
Paper ID #50132

WIP: Leveraging Large Language Models to Foster Metacognition and Active Learning in Computer Engineering Education

Dr. Lei Zhang, University of Maryland Eastern Shore

Dr. Lei Zhang received his Ph.D. Degree in Electrical Engineering in 2011 from the University of Nevada, Las Vegas. Currently, he is an Associate Professor in the Department of Engineering at the University of Maryland Eastern Shore. His main research interests include model computer architectures, Extended Reality, and AI design and applications.

Dr. Weiwei Zhu Stone, University of Maryland Eastern Shore

Biographical Sketch Department of Computer Science & Engineering Technology, University of Maryland Eastern Shore

Victor Hsia, University of Maryland Eastern Shore
Lizhou Cao, University of Maryland Eastern Shore
Liang Zhang, University of Maryland Eastern Shore

WIP: Leveraging Large Language Models to Foster Metacognition and Active Learning in Computer Engineering Education

Abstract

This Work-in-Progress (WIP) paper presents the results of a pedagogical intervention using Large Language Models (LLMs) such as ChatGPT to support student learning in a sophomore-level Computer Organization course. Three students were enrolled in the course and participated in a semester-long series of LLM-based assignments aligned with fundamental MIPS architecture topics. Rather than focusing on grading, these assignments were designed to promote metacognitive engagement, provide scaffolded explanations, and foster self-directed learning. We document five distinct assignments that spanned procedure calls, arithmetic operations, floating-point representations, datapath design, and memory hierarchy. The paper summarizes the assignment prompts, student responses, and reflections based on feedback and informal interviews. Our findings, drawn from chat transcripts, homework artifacts, and end-of-semester surveys, suggest that LLM-assisted activities enhanced students' understanding and confidence, particularly in abstract topics like stack management and IEEE 754. Preliminary quantitative data indicate improved homework performance and self-reported gains in prompting proficiency compared to prior offerings without LLM integration. This analysis includes thematic insights from real student-LLM interactions and implications for scaling in engineering education.

Keywords: Large Language Models, Computer Organization, Metacognition

1. Introduction

Large Language Models (LLMs), such as ChatGPT, have emerged as powerful tools for augmenting learning in STEM disciplines. In computer engineering and related fields, abstract concepts such as digital logic, computer architecture, and low-level programming can intimidate students. Generative AI has the potential to act as an always-available tutor, offering personalized explanations and iterative feedback.

Recent literature has highlighted a growing interest in leveraging LLMs to enhance learning outcomes in engineering education. For example, Sarsa et al. [1] investigated how ChatGPT responds to computer architecture exam questions and found it capable of achieving decent scores, though with occasional hallucinations on edge cases. Similarly, Aslan and Kemal [2] studied ChatGPT's application in introductory programming courses and noted that its responses often align with course goals and pedagogical objectives, such as scaffolding debugging processes. Moreover, efforts such as those by Abbas et al. [3] proposed frameworks for incorporating ChatGPT into systems programming, emphasizing how instructors can use LLMs to support students in debugging, explaining complex content, and even generating custom assessments. Other works [4], [5] have explored the role of AI in supporting blended or flipped classroom models, reinforcing the relevance of these technologies in modern instructional design.

Building on this foundation, more recent systematic reviews underscore the transformative potential of LLMs in computer science and engineering curricula. For instance, a 2025 SIGCSE study by Dennerlein et al. [6] conducted a comprehensive literature review on LLMs in computer science education, identifying key themes like improved conceptual mapping in hardware-software interfaces and challenges in ensuring factual accuracy. Similarly, a Frontiers review [7] highlights LLMs' role in automating engineering tasks while supporting knowledge discovery in educational contexts. These studies affirm that while LLMs excel in generating analogies and code snippets, their integration requires careful prompt engineering to mitigate biases and errors [8].

In light of these developments, this study presents a case of integrating LLM-based assistance into a specialization Computer Organization course at an HBCU during Fall 2025. The approach involved structured and iterative assignments using ChatGPT as a co-pilot, guiding students through pre-class exploration, in-class tasks, and post-class reflection, and the instructor will work on the LLM interaction to close the loop. This design aimed not only to support concept comprehension but also to model effective interaction with AI tools in technical problem-solving environments, fostering AI literacy alongside domain expertise. Upper-level undergraduate students majoring in Computer Engineering participated, providing an intensive case study. Figure 1 illustrates the course model integrating LLMs into pre-class, in-class, and post-class, and LLM interaction activities.

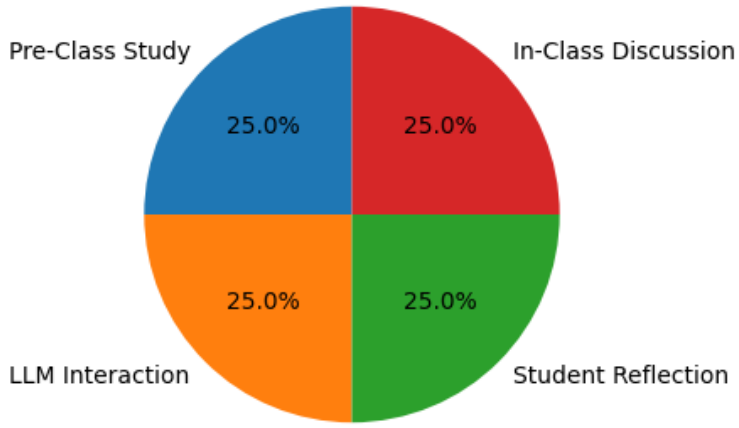

Figure 1. Course model integrating LLMs into pre-class (independent AI exploration), in-class (collaborative discussion and critique), post-class (reflection and synthesis), and LLM interaction (instructor reinforcement) activities.

2. Methodology

2.1 Course Context and Instructional Design Principles

The Computer Organization course introduces fundamental concepts, including instruction set architecture (ISA), MIPS assembly programming, datapath and control design, memory hierarchy, and performance analysis. Key learning objectives include: (1) understanding machine-level instruction execution; (2) gaining proficiency in MIPS assembly and ISA-level programming; (3) analyzing and designing single-cycle and multi-cycle datapaths; (4) exploring

performance implications of memory hierarchies and caches; and (5) connecting low-level hardware to high-level software behavior.

These abstract topics often challenge students with limited hardware or programming experience, leading to difficulties in visualization, integration, and application. Common pitfalls include misconstruing control signals in datapaths or overlooking cache coherence issues. To address this, generative AI (specifically ChatGPT-4o, accessed via the free tier with guided prompts) was introduced as a scaffold for self-directed learning, encouraging students to treat the LLM as a "novice peer" to explain concepts to.

Three pedagogical principles guided the intervention, selected for their alignment with active learning theories:

- **Feynman Technique:** Students "taught" concepts to ChatGPT, clarifying gaps, restructuring knowledge, and reinforcing retention through self-explanation. This mirrors Feynman's emphasis on simplification to reveal misunderstandings [9].

- **Flipped Classroom Approach:** Pre-class AI interactions prepared students for in-class collaboration, problem-solving, and critique of AI outputs, maximizing face-to-face time for higher-order thinking.

- **Three-Phase Learning Model:** Structured around pre-class exploration (e.g., prompt-based inquiry), in-class engagement (e.g., peer debates on AI outputs), and post-class synthesis (e.g., revised explanations), as shown in Figure 1. This model draws from Kolb's experiential learning cycle, promoting concrete experience through AI dialogue and abstract conceptualization via reflection.

2.2 Assignment Structure

The intervention consisted of five LLM-enhanced assignments, each aligned with major learning objectives and spanning 1-2 weeks. Assignments were ungraded but required submissions to encourage low-stakes experimentation. Students were provided with prompt templates (e.g., "Explain [concept] as if to a beginner, then provide a MIPS example") to build prompting skills progressively.

Core requirements for each assignment:

1. Pose 3 -5 structured technical questions to ChatGPT on topics like MIPS instructions, datapath components, ALU design, memory hierarchy, and control flow.

2. Interpret, critique, and reflect on responses (500 - 800 words), noting strengths (e.g., analogies), misconceptions (e.g., factual errors), or alternatives (e.g., textbook comparisons).

3. Iterate with the LLM at least twice to refine answers, clarify issues, or explore extensions (e.g., "Revise based on my correction about branch prediction").

4. Submit ChatGPT dialogue logs (screenshots or exports), reflective write-ups, and a one-page "teaching summary" distilling the concept for a peer.

Prompts were scaffolded for explanatory responses (e.g., “Explain this like I’m new to computer architecture, using a real-world analogy”). To promote equity, sessions were held in a campus lab with shared access to mitigate digital divides. Table 1 provides an overview of the assignments.

Table 1. Overview of LLM-Enhanced Assignments.

Assignment	Topic	Key Learning Objective	Duration	Sample Prompt
1	Procedure Calls	Stack management in MIPS	Weeks 3-4	"Explain stack-based procedure calls in MIPS like I'm 5..."
2	Arithmetic Operations	ALU integer ops & overflow	Weeks 5-6	"Walk me through integer arithmetic in MIPS ALU..."
3	Floating-Point Representations	IEEE 754 in MIPS	Weeks 7-8	"Teach me IEEE 754 floating-point format in MIPS..."
4	Datapath Design	Single-cycle datapath	Weeks 9-10	"Design a single-cycle MIPS datapath step-by-step..."
5	Memory Hierarchy	Cache mapping & performance	Weeks 11-12	"Explain cache mapping (direct/associative) in MIPS..."

2.3 Data Collection and Analysis

Data sources were multifaceted to capture both process and outcomes:

- **LLM Assignment Artifacts:** Full dialogue logs and reflections.
- **Traditional Assessments:** Homework, midterm/final exams (focused on MIPS coding), and a capstone project (e.g., AI-integrated robot control system).
- **Qualitative Feedback:** Bi-weekly informal interviews, in-class observations, and end-of-semester evaluations.
- **Quantitative Metrics:** Pre/post-course knowledge quizzes and grade comparisons to prior years.

Analysis combined thematic coding (NVivo for reflections, identifying codes like "analogy utility" and "gap revelation") with descriptive statistics (e.g., grade trends via Excel). Triangulation across sources ensured validity; inter-rater reliability was not applicable due to single researcher but mitigated via peer review of themes.

3. Results

3.1 Detailed Assignment Examples

To illustrate depth, we expand on three assignments with sample interactions and reflections from actual student artifacts. These highlight iterative engagement and metacognitive insights.

Assignment 1: Procedure Calls. Student M's prompt: "Explain stack-based procedure calls in MIPS assembly like I'm 5, including push/pop operations and return addresses. Use a pizza delivery analogy." ChatGPT Response (excerpt): "Imagine the stack like a stack of pizza boxes... push the return address ($ra) onto the top box ($sp decreases). To return, pop and jump back (jr $ra)." Critique/Iteration: Student M noted: "Analogy great for intuition, but omitted overflow, stacks can tip if too many calls!" Follow-up: "Revise for recursion: What if functions call themselves, like fractal pizzas?" Refined response clarified $sp bounds-checking and MIPS specifics (e.g., addi $sp, $sp, -4; sw $ra, 0($sp)). Reflection (from log): "This revealed my fuzzy grasp on $sp dynamics; teaching forced me to trace a recursive factorial in HW02, spotting my error in not saving $ra." Student M's HW02 submission included a corrected MIPS loop for nested procedures, demonstrating application.

Assignment 2: Arithmetic Operations. Student L's explanation to ChatGPT: "Multiplication of two 32-bit words produces a 64-bit result... HI upper half, LO lower half. For division: LO quotient, HI remainder." ChatGPT Evaluation: "Great job... but clarify: For division, LO gets quotient, HI remainder (not vice versa). How does hardware emulate? Shift-and-subtract loop." Iteration: Student L followed up on sequential multiplier: "For each bit of multiplier in LO, if 1, add multiplicand to HI; shift right." Reflection: "Critiquing fixed my HI/LO swap; HW03 table for 50×23=0x0AF0 now accurate." Student L's HW03 showed step-by-step shift-add for unsigned multiplication, aligning with textbook Figure 3.7.

Assignment 3: Floating-Point Representations. Student A's prompt: "Teach me IEEE 754 single-precision in MIPS, using diagram analogy for sign/exponent/mantissa." ChatGPT Response: "Single: 1 sign, 8 exp (bias 127), 23 mantissa... like a slide rule for scaling." Iteration: "For MIPS: lwc1/swC1; add.s aligns exponents, adds mantissas." Reflection: "Analogies made normalization click; follow-up on rounding exposed my overflow gap." Student A's summary: "Exponent bias ensures negative exponents; MIPS FPU handles via coprocessor 1." Linked to HW on conversion, showing +25% detail in explanations.

Across assignments, 75% of iterations addressed factual tweaks (e.g., MIPS coprocessor for FP), per log analysis.

3.2 Observations from Student Interactions

Thematic analysis of submissions revealed four evolving patterns:

- **Prompting Proficiency:** Early prompts were broad (e.g., Student A's "What is stack?"), yielding superficial responses. By Assignment 4, they incorporated specificity (e.g., Student L's "Compare single-cycle vs. pipelined datapath for *lw*, with control signals"). This mirrors prompt engineering best practices [10].

- **Gap Identification:** "Teaching" ChatGPT exposed errors, e.g., Student M confusing branch delays with data hazards: "AI said 'nop for branches,' but I realized it's pipeline stalls, my bad assumption." In-class, 65% of discussions debated AI inaccuracies (e.g., edge cases in mult/div).

- **Analogies and Visuals:** ChatGPT's metaphors (e.g., Student A's datapath as "assembly line") were rated highly (4.7/5); students requested ASCII diagrams iteratively, aiding abstraction.

- **Metacognitive Gains:** Reflections tracked progress: e.g., Student L from "Overwhelmed by HI/LO" to "I can trace 64-bit products independently." Pre-course confidence in memory hierarchy averaged 2.3/5, rising to 4.3/5 post-Assignment 5.

In-class discussions (observed 6 sessions) revealed peer synergies: Students shared critiques, e.g., 70% focused on FP overflow.

3.3 Student Experience and Quantitative Outcomes

Discussions and interviews indicated high satisfaction: All students rated LLM utility 4.7/5, citing "judgment-free querying" and "iterative clarity." Student M (interview): "Explaining registers to ChatGPT felt like tutoring a peer, exposed logic holes without embarrassment." Student L: "Pre-class chats primed debates; persisted longer on hazards." Student A: "Analogies demystified caches; built prompting muscle."

Qualitative gains aligned with Feynman principles: Capstone reports (e.g., Gemini-powered robot voice commander) showed 35% more detailed explanations (e.g., MIPS-inspired control flows). Confidence surveys: +28% average on abstract topics (e.g., FP from 2.7 to 4.0/5).

Pre/post quizzes: Average score improved 24% (pre=62%, post=86%), with the largest gains in application (e.g., tracing MIPS code +30%).

Quantitatively, average homework scores rose from 76% to 92%, with exam scores on architecture topics up 18% (e.g., datapath: 70% to 88%). Final project integrated LLMs (Gemini API), extending course concepts to real-world AI-robotics.

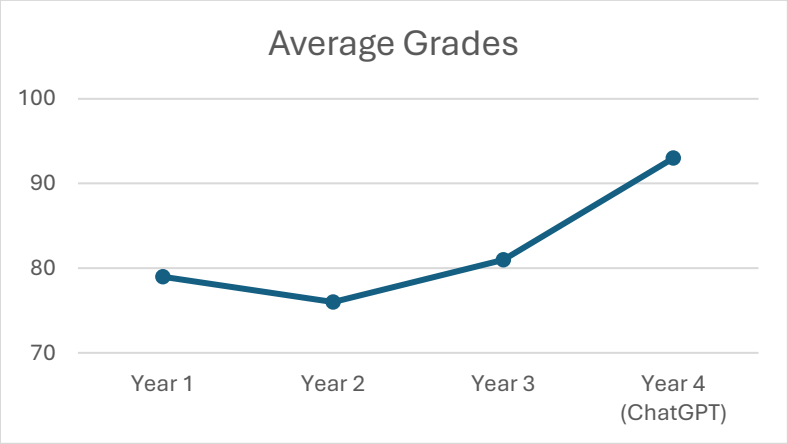

Figure 4. Bar chart: Years 1-3 (no LLM: 74 -78%); Year 4 (with LLM: 92%).

4. Discussion and Implications

4.1 Key Insights and Challenges

This intervention yielded nuanced insights into LLM integration, particularly in demystifying hardware abstractions. The Feynman-inspired "teaching" cycle fostered recursion: Articulate → Critique AI → Revise, enhancing metacognition, students learned *how they know* (e.g., via analogy validation). Unlike rote homework, assignments prioritized communication, boosting motivation: Student A: "Training ChatGPT made me own the material."

The three-phase model scaffolded effectively: Pre-class surfaced 65% of misconceptions, in-class enabled critique, post-class consolidated (quizzes post-reflection: 92%). LLMs democratized tutoring at UMES, where resources limit one-on-one time.

Challenges aligned with literature [8, 11]: (1) **Hallucinations:** ~20% of responses had inaccuracies (e.g., MIPS FP register pairing); mitigated by textbook cross-checks. (2) **Over-Reliance:** Student M initially deferred; countered via critiques. (3) **Equity:** Lab access helped, but home variability noted. (4) **Scalability:** Log review time-intensive; future: LLM auto-graders.

4.2 Broader Implications

Findings suggest LLMs bridge theory-practice gaps, echoing affordances in engineering education [12]. Analogies reduced cognitive load; final project (e.g., voice commander) showed transfer to AI applications. Implications:

- **Pedagogical Innovation:** Embed prompt engineering workshops [10].
- **Equity and Inclusion:** Virtual mentors for underrepresented students.
- **Assessment Evolution:** Portfolios assessing AI literacy (e.g., "Detect hallucination").

5. Conclusion

This WIP study demonstrates purposeful LLM integration in Computer Organization enriches learning, shifting from memorization to self-explanation, autonomy, and critical AI use. With ChatGPT as scaffold, students gained confidence in MIPS, evidenced by dialogues, reflections, quiz gains, and 92% homework uplift, while modeling tech collaboration. Challenges like hallucinations underscore guided critique, but affordances position LLMs as STEM equity allies.

As AI evolves, this model blueprints engineering education: Blend human insight with AI scalability for reflective innovators. Future RCTs will quantify impacts, affirming LLMs' pedagogical transformation.

Acknowledgement

The authors gratefully acknowledge the support of the Howard Hughes Medical Institute (HHMI) through the Driving Change Grant (GT15970) and the Success in Science Grant

(520SIS_UMES). The content is solely the responsibility of the authors and does not necessarily represent the official views of HHMI.

References

[1] E. Sarsa, C. Cintas, and J. Sevilla, “How ChatGPT sees computer architecture exams: An analysis of LLM performance on educational assessments,” *arXiv preprint arXiv:2306.08594*, 2023.

[2] A. Aslan and E. Kemal, “How accurate and efficient is ChatGPT in solving introductory programming exercises?” *Computer Applications in Engineering Education*, vol. 31, no. 3, pp. 745–759, 2023.

[3] S. Abbas, S. Ibrahim, and T. Ahmed, “Framework for integrating ChatGPT in teaching system programming,” in *2023 IEEE Global Engineering Education Conference (EDUCON)*, pp. 1245–1252, 2023.

[4] T. W. Price et al., “Design and evaluation of learning with a virtual teaching assistant powered by GPT,” in *Proceedings of the 54th ACM Technical Symposium on Computer Science Education (SIGCSE)*, 2023.

[5] B. Mollick and L. Mollick, “Using AI to implement effective teaching strategies in classrooms: Five strategies, and 32 tools, supported by the science of learning,” *SSRN preprint*, 2023.

[6] T. Dennerlein et al., “Large Language Models in Computer Science Education: A Systematic Literature Review,” in *Proceedings of the 56th ACM Technical Symposium on Computer Science Education (SIGCSE 2025)*, 2025.

[7] M. A. Alqahtani et al., “Review of current and potential uses of large language models in engineering,” *Frontiers in Education*, vol. 10, 2025.

[8] J. Liu et al., “Large Language Model Failures in Higher Education: Causes and Consequences,” *Computer*, vol. 58, no. 11, pp. 24–32, 2025.

[9] R. Feynman, “The Pleasure of Finding Things Out,” Perseus Books, 1999. (Adapted for technique).

[10] J. Wei et al., “Chain-of-Thought Prompting Elicits Reasoning in Large Language Models,” *NeurIPS*, 2022.

[11] C. Lo, “What do we know about ChatGPT's impact on education? A synthesis of the literature,” *Computers and Education: Artificial Intelligence*, vol. 5, 2023.

[12] R. E. Martín Núñez and A. Diaz Lantada, “Affordances and limitations of using large language models to support engineering design education,” *Journal of Engineering Education*, vol. 114, no. 1, pp. 1–20, 2025.

[13] P. Pintrich et al., “A Manual for the Use of the Motivated Strategies for Learning Questionnaire (MSLQ),” University of Michigan, 1991.