Application of Cloud-Based Digital Prototyping Systems for Developing Hardware Logic Skills

Andrii Lobatskyi[1], Pavlo Savaryn[1], Volodymyr Podolyak[2], Mykhailo Lepkiy[2] and Vitalii Kabak[1]
[1]*Department of Digital Educational Technologies, Faculty of Digital, Educational and Social Technologies, Lutsk National Technical University, Lvivska Str. 75, 43018 Lutsk, Ukraine*
[2]*Department of Tourism and Hotel and Restaurant Business, Lutsk National Technical University, Lvivska Str. 75, 43018 Lutsk, Ukraine*
a.lobatskyi@lntu.edu.ua, savaryn.pasha@lntu.edu.ua, v.podoliak@lutsk-ntu.com.ua, m.lepkiy@lutsk-ntu.com.ua, v.kabak@lutsk-ntu.com.ua

Keywords: Cloud-Based Prototyping, Digital Model, Digital-Twin-Inspired Workflow, EDA Tools, Computer Architecture Education, Virtual Verification, Hardware Logic, Tinkercad Circuits.

Abstract: The abstraction gap between theoretical Boolean logic and physical circuit implementation remains a persistent pedagogical challenge in Computer Architecture education. This paper investigates a simulation-first workflow that uses Tinkercad Circuits as a cloud-based digital model environment for pre-physical verification of hardware logic designs. The proposed workflow follows a four-phase iterative process: logic design specification, virtual prototyping, virtual verification with integrated debugging tools, and physical breadboard implementation. A pilot implementation with 20 first-year computer engineering students showed promising descriptive outcomes: 18 students, or 90%, completed all assignments with correct functionality; average laboratory time decreased from 120 minutes to 80-90 minutes per assignment; and instructor assistance shifted from mechanical troubleshooting to conceptual analysis of logic behavior, timing, and design alternatives. Because the study involved a single cohort without a parallel control group, these findings should be interpreted as preliminary evidence of feasibility and pedagogical potential rather than statistically conclusive proof of superiority over breadboard-first instruction.

1 INTRODUCTION

Contemporary computer engineering curricula face a fundamental pedagogical challenge: the abstraction gap between conceptual understanding of digital logic and practical implementation of hardware circuits [1], [2]. Students readily master truth tables, Boolean algebra, and logic gate theory in abstract form, yet encounter significant difficulties when translating these concepts into functional physical circuits. This disconnect manifests as frequent wiring errors, component misidentification, and an inability to diagnose circuit malfunctions-skills essential for professional competence in hardware design [3].

The Industry 4.0 paradigm has fundamentally transformed engineering practice, establishing digital prototyping and virtual commissioning as standard methodology for hardware development [4], [5]. Modern industrial workflows leverage digital twin technology-virtual replicas of physical systems that enable simulation, verification, and optimization before physical fabrication [6]. However, traditional computer architecture pedagogy remains anchored to breadboard-first methodologies that lack the iterative design verification cycles characteristic of contemporary engineering practice.

1.1 The Pedagogical Problem: Cognitive Overload in Novice Hardware Education

Introductory hardware laboratories present novice students with simultaneous cognitive demands: understanding abstract logic principles, managing physical component constraints, troubleshooting invisible electrical phenomena, and manipulating unfamiliar tools. Breadboard prototyping introduces systematic error patterns-component misplacement, incorrect polarity, loose connections, and misinterpreted topology-that consume instructional time on mechanical troubleshooting rather than conceptual development [3].

Electronic Design Automation (EDA) tools constitute the professional standard for circuit design and verification. High-fidelity commercial platforms such as Altium Designer and Proteus offer comprehensive simulation capabilities but present prohibitive entry barriers for novice learners: steep learning curves, complex licensing, and sophisticated feature sets that overwhelm introductory students. Cloud-based rapid prototyping environments-exemplified by Tinkercad Circuits-occupy an optimal middle ground, providing authentic simulation fidelity with minimal cognitive overhead for mastering the interface.

1.2 Research Objectives and Contribution

This paper investigates whether a simulation-first pedagogical workflow utilizing cloud-based digital twin environments can reduce implementation errors, accelerate concept-to-hardware translation, and shift student attention from mechanical assembly to logical analysis in introductory computer architecture education. We position Tinkercad Circuits not merely as educational software but as an accessible rapid-prototyping platform for creating digital models of hardware logic circuits within a digital-twin-inspired pedagogical workflow. We use the term digital-twin-inspired workflow in a pedagogical sense. In its strict industrial definition, a digital twin requires bidirectional, often real-time, data exchange between a physical object and its virtual representation, which is not implemented in this study. Accordingly, the Tinkercad Circuits prototype is treated as a simplified but pedagogically adequate digital model for pre-physical verification rather than as a runtime mirror of an existing physical artifact.

Our contribution is threefold:

1) articulation of a systematic four-phase digital prototyping workflow adaptable to diverse hardware logic curricula;
2) pilot implementation results demonstrating improved task completion rates, reduced laboratory time, and qualitative shift in assistance request patterns;
3) identification of cognitive mechanisms-particularly visualization of hidden electrical states and safe failure environments-that account for observed improvements.

2 LITERATURE REVIEW

2.1 Digital Twin Technology in Engineering Education

Digital twin technology has emerged as a transformative pedagogical instrument across engineering disciplines. Recent studies report that digital twin-based learning can improve efficiency and performance relative to conventional instruction [4], [5]. Educational digital twins provide cost-effective alternatives to physical laboratories while maintaining high simulation fidelity, addressing resource constraints in engineering programs [6], [11].

2.2 Simulation-First Versus Hands-On Laboratory Instruction

The comparative effectiveness of simulation-based versus hands-on laboratory instruction remains an active research question. Taher et al.'s comprehensive study of electronics education revealed a critical nuance: simulation environments are most effective when used as pre-physical scaffolding rather than as complete replacements for hands-on practice [7]. Students exposed to hybrid instructional sequences-simulation followed by physical implementation-demonstrated superior configuration speed and troubleshooting efficiency compared to either approach in isolation.

Kadir's research on virtual laboratories established positive transfer of training from virtual to physical environments, where students using virtual simulations showed significantly higher proficiency ($p < 0.05$) in subsequent physical implementation and troubleshooting compared to traditional groups [8]. These findings support a simulation-first rather than simulation-only pedagogical model, where virtual prototyping serves as conceptual scaffolding before tactile implementation [11].

2.3 Cloud-Based Tools for Hardware Education

Emerging research on Tinkercad Circuits validates its efficacy as a pedagogical tool for teaching hardware logic. Studies document learning outcome improvements ranging from 11.5% to 25% among electronics and computational thinking assessments [9], [10]. Critically, cloud-based platforms provide a visual representation of electrical states-current flow, voltage levels, logic signal propagation-that remain invisible in physical breadboard implementations without specialized instrumentation [3]. This visualization capacity addresses a primary cognitive barrier in novice hardware education: the inability to perceive abstract electrical phenomena as tangible, observable processes. Real-time color-coded wire highlighting, logic level indicators, and virtual instrumentation enable students to construct accurate mental models of circuit behavior through direct observation rather than hypothetical reasoning.

2.4 Error Patterns in Breadboard Construction

DesPortes' analysis of novice breadboard construction identified recurring failure modes: component misplacement (incorrect IC orientation, wrong pin rows), wire-related errors (omissions, faulty connections, inadvertent disconnections during handling), and topology misunderstandings (failure to comprehend internal breadboard connection patterns) [3]. Novice students report these errors as "intimidating" and "difficult to diagnose," with error identification consuming disproportionate instructional time. The cognitive load imposed by simultaneous management of spatial constraints, tactile precision, and hidden electrical topology diverts working memory resources from the primary learning objective: understanding circuit logic and functionality.

3 METHODOLOGY

3.1 Pedagogical Context and Course Structure

This pilot implementation was conducted within the "Information Coding and Computer Architecture" course, a first-year required module for Computer Engineering students. The course curriculum covers combinational logic circuits (logic gates, half-adders, full-adders, decoders), sequential circuits (flip-flops, registers, counters), and basic microarchitecture concepts. Traditional course delivery employed breadboard-first laboratory sessions in which students constructed circuits directly from the instructors' schematic diagrams.

3.2 Study Design and Participants

The simulation-first workflow was implemented with one laboratory cohort N = 20 during the 2025-2026 academic year. Students had completed prerequisite coursework in digital logic fundamentals but had minimal prior experience with physical circuit construction or simulation tools. All participants provided informed consent for educational data collection in accordance with institutional review board protocols.

The laboratory classes were structured in the form of 2-hour blocks, which gradually became more complex: (1) two-input logic gates (AND, OR, NOT, XOR), (2) half-adder circuits, (3) full-adder implementation, (4) 4-to-1 multiplexer, (5) 3-to-8 decoder, and (6) 4-bit synchronous counter using D flip-flops. Each assignment required students to complete the four-phase workflow described below before receiving instructor verification.

3.3 The Four-Phase Digital Prototyping Workflow

We structured laboratory activities as an iterative engineering workflow analogous to professional hardware development cycles in Industry 4.0 contexts:

Phase 1: Logic Design Specification. Students receive functional requirements expressed as truth tables or Boolean expressions. Rather than immediately translating specifications into physical components, students first perform logic minimization using Karnaugh maps or Boolean algebra to determine optimal gate-level implementations. This phase emphasizes algorithmic thinking and design optimization without hardware constraints, typically requiring 10-15 minutes per assignment.

Phase 2: Digital model creation (Virtual Prototyping). Using the Tinkercad Circuits cloud platform, students construct virtual circuit prototypes by selecting components from a library that accurately model physical ICs (logic gates, flip-flops), discrete components (resistors, LEDs, switches), power supplies, and wiring. The platform provides realistic visual representations, including correct pinout diagrams and electrical characteristics, supporting consistency between the simulated model and the subsequent physical implementation Figure 1.

This phase serves dual pedagogical functions: (1) reinforcement of component identification and specification interpretation skills through visual library browsing, and (2) initial circuit topology planning in a consequence-free environment where wiring errors produce no physical damage or time penalties beyond immediate correction.

Phase 3: Virtual Verification and Debugging. Students activate simulation mode to verify circuit functionality through systematic testing. Tinkercad provides real-time visualization capabilities critical for novice comprehension: active circuit traces are highlighted in distinct colors based on logic states (red for HIGH, blue for LOW, and gray for undefined), virtual LED indicators display outputs, and integrated multimeters enable voltage/continuity measurements Figure 2.

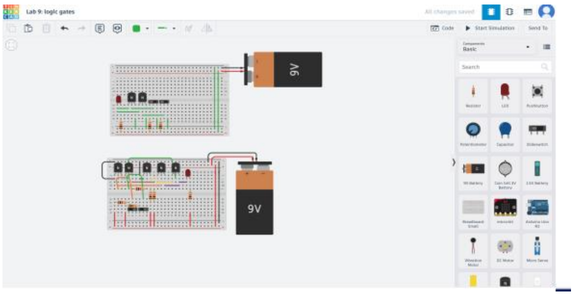
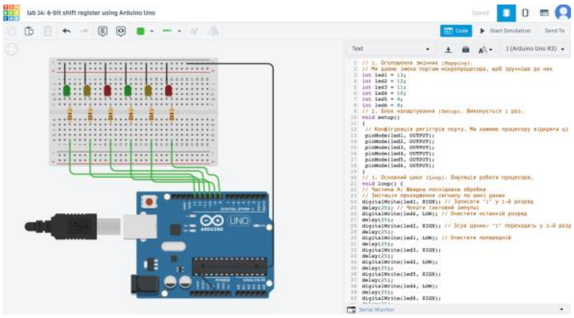


Figure 1: Cloud-based rapid prototyping environment in Tinkercad Circuits demonstrating virtual prototyping of combinational logic gates (AND, OR, XOR) with realistic component models, breadboard topology visualization, and power supply configuration.



```
// 1. Оголошення змінних (Mapping).
// Ми даємо імена портам мікропроцесора, щоб зручніше до них
int led1 = 13;
int led2 = 12;
int led3 = 11;
int led4 = 10;
int led5 = 9;
int led6 = 8;
// 2. Блок налаштування (Setup). Виконується 1 раз.
void setup()
{
 // Конфігурація регістрів порту. Ми кажемо процесору відкрити ці
 pinMode(led1, OUTPUT);
 pinMode(led2, OUTPUT);
 pinMode(led3, OUTPUT);
 pinMode(led4, OUTPUT);
 pinMode(led5, OUTPUT);
 pinMode(led6, OUTPUT);
}
// 3. Основний цикл (Loop). Емуляція роботи процесора.
void loop() {
// Частина А: Швидка послідовна обробка
// Імітація проходження сигналу по шині даних
digitalWrite(led1, HIGH); // Записати "1" у 1-й розряд
delay(25); // Чекати тактовий імпульс
digitalWrite(led6, LOW); // Очистити останній розряд
delay(25);
digitalWrite(led2, HIGH); // Зсув даних: "1" переходить у 2-й розр
delay(25);
digitalWrite(led1, LOW); // Очистити попередній
delay(25);
digitalWrite(led3, HIGH);
delay(25);
digitalWrite(led2, LOW);
delay(25);
digitalWrite(led5, HIGH);
delay(25);
digitalWrite(led4, LOW);
delay(25);
digitalWrite(led6, HIGH);
```

Figure 2: Advanced sequential logic implementation: 6-bit shift register using Arduino Uno microcontroller with visual LED output array. Color-coded wiring demonstrates organized circuit topology and real-time state visualization during simulation, illustrating progression from simple gates to programmatically controlled sequential circuits.

This phase shifted lab work from passive diagram-following to active experimentation. Tinkercad's visual feedback loop enabled interactive debugging (e.g., toggling inputs and observing real-time trace changes), reducing fear of component damage and supporting "what-if" testing. By making hidden electrical states observable-signal propagation, floating inputs, and shorts-students could correct errors immediately without physical instrumentation.

Phase 4: Physical Implementation. After verifying functionality in simulation, students build the circuit on a breadboard using their validated digital model rather than abstract schematics. They replicate component placement, wiring, and power connections from Tinkercad, which serves as assembly reference. As a result, hands-on work becomes a controlled translation of a pre-tested design instead of simultaneous construction and troubleshooting, reducing cognitive load.

3.4 Data Collection and Analysis Methods

We collected both quantitative performance metrics and qualitative observational data across all six laboratory assignments. Quantitative metrics included: (1) task completion status (functional circuit passing all specified tests), (2) time from session start to successful deployment, (3) number and category of instructor assistance requests, and (4) first-attempt success rate (circuit functions correctly upon initial physical assembly without rewiring). Teaching assistants logged assistance requests using a structured taxonomy distinguishing basic mechanical issues (component orientation, wire connections, power supply) from conceptual questions (logic analysis, timing behavior, circuit modifications).

Qualitative data collection employed semi-structured observation protocols during laboratory sessions, noting student debugging strategies, error self-correction behaviors, and expressed confidence levels. Post-course anonymous surveys solicited student perceptions of workflow effectiveness, cognitive load experiences, and preference comparisons between simulation-first and traditional approaches.

4 RESULTS

4.1 Task Completion and Implementation Success

Among the 20 students who completed the simulation-first workflow, 18 students, or 90%, successfully implemented all six laboratory assignments, with functionality verified through systematic testing. This exceeds typical outcomes from prior breadboard-first semesters, where success rates for complex tasks (e.g., multiplexers, counters) often ranged 60-70% within the allotted lab time.

The two students who struggled during physical implementation still verified correct designs in Tinkercad but faced breadboard assembly issues-mainly incomplete IC pin insertion causing intermittent contacts. They quickly isolated these mechanical faults by comparing physical behavior with the known-good simulation, indicating effective transfer of the virtual verification approach to the physical setup.

4.2 Time Efficiency Analysis

Laboratory time-allocation analysis showed substantial efficiency gains versus traditional instruction. In prior semesters, students typically required the full 120 minutes, with 30-40% failing to complete functional implementations within the session and needing additional office hours.

With the simulation-first workflow, students finished in 80-90 minutes on average (25-33% reduction). Time was distributed across phases: Phase 1 (15-20 min), Phase 2 (20-25 min), Phase 3 (~20 min), Phase 4 (25-30 min). Because designs were pre-verified in simulation, physical assembly shifted from troubleshooting wiring errors to more controlled execution focused on tactile accuracy, guided by students' digital twins.

Overall, this provided ~30 minutes of additional learning time per lab; across six assignments, this equals 3-4 hours of recovered instructional time per student, reallocated from troubleshooting to design variations and conceptual analysis.

4.3 Qualitative Shift in Assistance Request Patterns

The most pedagogically significant finding was the shift in instructor assistance requests. Teaching assistants kept structured logs classifying questions as mechanical/procedural (component identification, wiring, power, polarity) or conceptual/analytical (logic behavior, truth-table reasoning, timing, design modifications).

In breadboard-first labs, mechanical requests dominated: students needed help with wiring errors, IC orientation, loose connections, and fault location; instructors estimated 70-80% of support time went to low-level troubleshooting. Under the simulation-first workflow, requests shifted toward reasoning and design.

Examples from the logs illustrate the change. Traditional questions: "Which direction should this IC face?", "Where do I connect ground?", "Why isn't my LED lighting up?" Simulation-first questions: "Why does this gate configuration produce this output pattern?", "How would I extend this 4-bit counter to 8 bits?", "What happens if I change the clock frequency?" Overall, student attention moved from mechanical assembly to logical analysis and system-level thinking.

5 DISCUSSION

5.1 Visualization of Hidden Electrical States

The primary mechanism behind the improved outcomes is state visualization: in breadboard labs, voltage levels, current flow, and logic states are largely invisible without instrumentation, forcing students to rely on assumptions rather than observation. Tinkercad makes these states observable through color-coded logic traces, voltage probes, and signal-path highlighting, so students can empirically track propagation from inputs through gates to outputs (e.g., LEDs). This directly reduces novice diagnostic barriers documented in hardware education [3] and supports faster fault isolation using concrete checks such as: "Is this wire connected?", "Is power reaching the IC?", "What logic level is this node?", and "Why is the output incorrect?"

5.2 Safe Failure Environment and Iterative Learning

Traditional breadboarding can penalize experimentation: wiring mistakes may damage components, create safety risks, or require instructor intervention, which encourages risk-averse behavior and reduces independent hypothesis testing. Virtual prototyping removes these material consequences, providing immediate error/short-circuit indicators and allowing iterative "what-if" debugging without damage. This safe-to-fail setting supports an engineering mindset-reating errors as diagnostic feedback-and plausibly explains the observed shift from mechanical fault-location questions toward conceptual analysis and more independent troubleshooting during physical assembly [4], [5].

5.3 Cognitive Load Reduction Through Phased Separation

Traditional breadboard-first instruction imposes simultaneous cognitive demands: students must concurrently manage abstract logic concepts, physical component constraints, spatial wire routing, tactile precision, and hidden electrical topology. This cumulative cognitive load overwhelms working memory capacity, particularly for novice learners, resulting in both conceptual and mechanical errors [3].

The four-phase workflow strategically separates these cognitive demands across distinct activities. Phase 1 isolates pure logic design without hardware constraints. Phase 2 addresses component identification and specification interpretation in a visual interface without time pressure. Phase 3 focuses exclusively on logic verification and debugging with visible state feedback. Phase 4 becomes a mechanical translation of the pre-verified design, requiring only spatial reasoning and tactile skill without simultaneous logic analysis.

This phased separation enables students to develop competency in each skill domain independently before integration. By the time students approach physical breadboards, they have already verified logic correctness and established mental models of expected behaviour. However, the workflow should be understood as simulation-first rather than simulation-only. Tinkercad Circuits does not fully reproduce contact resistance, loose breadboard connections, component tolerances, or voltage instability under load. Therefore, Phase 4 should include brief hardware validation-checking power/ground continuity, measuring selected nodes, and discussing simulation-hardware discrepancies-o prevent an “illusion of simplicity” and distinguish logical correctness from physical reliability.

5.4 Alignment with Industry 4.0 Engineering Practice

Beyond immediate learning outcomes, positioning cloud-based prototyping as a digital-twin-inspired methodology yields benefits for professional identity formation. Contemporary hardware development universally employs simulation-before-fabrication workflows: digital verification using CAD tools and SPICE simulation precedes printed circuit board manufacturing, FPGA design undergoes extensive simulation before synthesis, and embedded systems utilize hardware-in-the-loop testing before production deployment [4], [6].

By structuring laboratory instruction around authentic professional workflows-design specification, digital model creation, virtual verification, physical implementation-students internalize industry-standard engineering practices from introductory coursework. This alignment prepares students for workplace expectations while demonstrating that educational activities reflect authentic professional competencies rather than artificial academic exercises.

6 CONCLUSIONS

The simulation-based workflow, tested as a pilot implementation with one cohort of first-year students N = 20, produced promising descriptive outcomes: 90% task completion and an observed 25-33% reduction in laboratory time. Given the limited sample size and the absence of a parallel control group, these results should be interpreted as preliminary evidence of feasibility and pedagogical potential rather than statistically conclusive proof of superiority over breadboard-first instruction. Reducing mechanical troubleshooting to 80-90 minutes freed up time for higher-level learning and helped bridge the gap between abstract Boolean logic and computer architecture practice.

The observed effect can be explained by three mechanisms: (1) visualization of hidden electrical states, (2) a safe environment for errors that supports exploratory learning without material/psychological risks, and (3) reduced cognitive load by separating logical design, virtual verification, and physical implementation into separate stages.

Cloud-based modeling is not considered a replacement for hands-on work, but rather a preliminary virtual verification stage that increases the efficiency of subsequent physical implementation, while final breadboard validation remains necessary. This hybrid approach (digital model → physical implementation) is consistent with Industry 4.0 practices and facilitates the transition of beginners from theory to circuit design.

Based on our pilot implementation experience, we recommend the following for instructors who use simulation-oriented workflows: (1) clearly position Tinkercad as an introductory EDA environment, not a “toy simulator,” to create an authentic engineering context; (2) require students to document the results of virtual verification (screenshots, test case tables) before approving physical implementation to ensure that steps 2-3 are actually performed, rather than superficial modeling; (3) allocate 40-50% of lab time to virtual prototyping steps, leaving the rest of the time for physical assembly, basic voltage checks, simulation-hardware comparison, and conceptual discussion, rather than basic troubleshooting; and (4) use the freed-up time to develop design options that promote a creative approach to problem solving, rather than mechanically reproducing the provided diagrams.

7 FUTURE WORK

As engineering education increasingly addresses Industry 4.0 workforce demands, cloud-based digital prototyping offers an accessible way to modernize hardware logic instruction without costly equipment. Within this pilot implementation, the workflow suggests that free cloud tools may support learning effectiveness, task completion, and pedagogical focus while familiarizing students with professional engineering practices. However, because the results come from a single cohort N = 20 without a parallel control group, they should be interpreted as exploratory rather than statistically conclusive. Future studies should test these trends with larger samples, control groups, and inferential statistical analysis. Such work would clarify the scalability of simulation-first approaches in computer engineering curricula.

REFERENCES

[1] C. Qian, S. Chen, and A. Goodney, “Bridging simulation and hardware: A holistic platform for teaching in computer architecture and operating systems,” in Proceedings of the 56th ACM Technical Symposium on Computer Science Education v. 2 (SIGCSE TS 2025), Pittsburgh, PA, USA, 2025, pp. 1591-1592, [Online]. Available: https://doi.org/10.1145/3641555.3705135.

[2] C. Mirolo, C. Izu, V. Lonati, and E. Scapin, “Abstraction in Computer Science Education: An Overview,” Informatics in Education, vol. 20, no. 4, pp. 615-639, 2021, [Online]. Available: https://doi.org/10.15388/infedu.2021.27.

[3] K. DesPortes et al., “BitBlox: A redesign of the breadboard,” in Proceedings of the 15th ACM Conference on Interaction Design and Children (IDC ’16), Manchester, United Kingdom, 2016, pp. 203-212, [Online]. Available: https://doi.org/10.1145/2930674.2930708.

[4] M. Pajpach, R. Pribiš, P. Drahoš, E. Kučera, and O. Haffner, “Methodology and pilot implementation of modular educational-development platform for interoperable digital twin,” Scientific Reports, vol. 15, art. no. 28020, 2025, [Online]. Available: https://doi.org/10.1038/s41598-025-13579-y.

[5] J. S. Ortiz, E. K. Quishpe, G. X. Sailema, and N. S. Guamán, “Digital twin-based active learning for industrial process control and supervision in industry 4.0,” Sensors, vol. 25, no. 7, art. no. 2076, 2025, [Online]. Available: https://doi.org/10.3390/s25072076.

[6] M. A. Hazrat, N. M. S. Hassan, A. A. Chowdhury, M. G. Rasul, and B. A. Taylor, “Developing a skilled workforce for future industry demand: The potential of digital twin-based teaching and learning practices in engineering education,” Sustainability, vol. 15, no. 23, art. no. 16433, 2023, [Online]. Available: https://doi.org/10.3390/su152316433.

[7] M. T. Taher and A. S. Khan, “Effectiveness of simulation versus hands-on labs: A case study for teaching an electronics course,” in Proceedings of the 122nd ASEE Annual Conference & Exposition, Seattle, WA, USA, 2015, art. no. 26.582, [Online]. Available: https://doi.org/10.18260/p.23920.

[8] A. Kadir, “Effectiveness of virtual laboratory utilization in improving students’ science process skills,” Ta’dib, vol. 26, no. 2, pp. 355-366, 2023, [Online]. Available: https://doi.org/10.31958/jt.v26i2.10913.

[9] Asbendri BL, S. Anori, I. P. Dewi, and Efrizon, “Exploring the impact of Tinkercad-assisted learning on student performance in industrial electronics subject,” Journal of Hypermedia & Technology-Enhanced Learning (J-HYTEL), vol. 2, no. 2, pp. 134-148, 2024, [Online]. Available: https://doi.org/10.58536/j-hytel.v2i2.124.

[10] C. Tüysüz, N. C. Bodur, and I. Ugulu, “Tinkercad Circuits Platform-Based Learning Experiences of Gifted Students in the Emergency Distance Education Process,” Journal of Advanced Academics, vol. 35, no. 2, pp. 329-356, 2024, [Online]. Available: https://doi.org/10.1177/1932202X241230589.

[11] O. Hulai, V. Shemet, I. Moroz, P. Savaryn, and V. Kabak, “Using the Labster virtual laboratory to study chemistry at a technical university,” Information Technologies and Learning Tools, vol. 102, no. 4, pp. 95-107, 2024, [Online]. Available: https://doi.org/10.33407/itlt.v102i4.5737.