International Journal of Science and Research (IJSR)
ISSN: 2319-7064
Impact Factor 2025: 7.089

Exploring Students' Experiences and Learning Outcomes of Python through BBC Micro:bit in the Age of Generative AI: A Mixed-Methods Classroom Action Research Study

A. Shikha Shah

PGT Computer Science, Tagore International School, East of Kailsh New Delhi-110065, India

Abstract: ***In the realm of programming education, the rise of Generative Artificial Intelligence (GenAI) tools like ChatGPT, Gemini, and GitHub Copilot has revolutionized the way students learn. Programming education has undergone a transformation thanks to Generative Artificial Intelligence (GenAI) tools such as ChatGPT, Gemini, and GitHub Copilot, which allow students to create functioning Python programs in mere moments. These tools offer excellent support, but can lead to superficial learning, reduced computational thinking, less debugging of code, and less creativity. This study explores the possibilities of using physical computing with BBC Micro:bit to address these challenges in the CBSE Class XI Computer Science curriculum. The study followed a classroom action research convergent mixed methods approach with 40 Senior secondary students for 40 days of intervention. Through hands-on projects, students were introduced to the concepts of Python: variables, data types, conditional statements, loops, functions, events, sensors, LED displays, music and radio communication. Amongst the projects were a Digital Dice, Rock-Paper-Scissors Game, Emotion Badge, Earthquake Monitor, Water Reminder, Footstep Tracker and an Alzheimer's Patient Tracker. The quantitative survey results showed that 94.7% of students found physical computing more engaging than traditional programming, while 84.2% noted that they had better logic-building skills when using physical computing. The qualitative thematic analysis provided evidence that students preferred hardware-based learning to software-based because it is interactive, visible, and meaningful, and that project-based learning allows for creativity and independence, while debugging is challenging but rewarding. The results indicate that using physical computing improves engagement, confidence, creativity and computational thinking, and decreases dependence on AI-generated solutions. The study suggests that project-driven learning based on actual hardware can continue to foster learner agency and critical thinking even in the era of AI-supported coding, a vital aspect of secondary computer science education.***

Keywords: Generative Artificial Intelligence, Python Programming, BBC Micro:bit, Physical Computing, Computational Thinking, Classroom Action Research, Qualitative Research, Project-Based Learning, Computer Science Education, Experiential Learning, CBSE, Student Engagement

1. Introduction

From ChatGPT to Gemini and GitHub Copilot, the swift advancement of generative AI (GenAI) tools has reshaped how students acquire programming skills.Tools like ChatGPT, Gemini, and GitHub Copilot are revolutionizing the way students learn to program. It is now possible to create syntactically correct Python programs in seconds with a single prompt. These new technologies offer very new ways to learn about coding solutions and explanations, but bring into focus important questions about shallow learning, missing opportunities for computational thinking, little opportunities for debugging, and limited creativity. This creates a pressing challenge for educators: how to create learning spaces that encourage active learning, problem-solving, and critical thinking while maintaining the advantages of AI-powered coding.

CBSE senior secondary (X) has made Python as the main language of programming in the Indian context. Typically, programming education focuses on syntax and solving text-based problems; however, this instruction often leads to an abstract learning experience, remote from real world applications. This approach may lead to further degradation of deep learning in the age of AI, where students might simply accept solutions without thinking for themselves or learning to tackle problems with persistence.

A promising pedagogical approach is physical computing, in which programs are made concrete by connecting programs to outputs in physical objects. The BBC Micro:bit enables students to control LEDs, sense motion, interpret input from sensors, and send signals, making coding an interactive, hands-on experience. Physical computing is an opportunity for learners to write code and see, test, debug, and refine in real-world situations. This method promotes experimentation, creativity, and computational thinking, and lessens reliance on AI-generated solutions.

This study explores the use of physical computing in Python teaching for CBSE class 11 Computer Science students. The research is guided by the use of project-based activities, including the creation of a Digital Dice, Earthquake Monitor, and Alzheimer's Patient Tracker, which will help capture the benefits of hardware-based programming in terms of enhanced engagement, logic development, creativity, confidence, and independent problem solving. The study is conducted using a mixed method classroom action research design, which involves both quantitative data from questionnaires and qualitative data from reflection. The results are expected to aid in the current discussion surrounding the use of AI in education and to suggest a pedagogy that could be applied in secondary schools to develop computational thinking skills in students in the future, which is physical computing.

2. Literature Review

- **Computational Thinking**: CT is a foundational skill for problem-solving in programming (Wing, 2006).
- **Physical Computing**: Hardware-based learning fosters engagement and creativity (Yurdakök & Kalelioğlu, 2024).
- **AI in Programming Education**: AI tools support syntax but risk shallow learning (Smith et al., 2024).

3. Methodology

- **Design**: Convergent mixed-methods classroom action research.
- **Participants**: 40 CBSE Class XI students.
- **Duration**: 40 days.
- **Intervention**: Python concepts taught via Micro:bit projects.
- **Data Collection**: Surveys (quantitative), reflections, journals, artefacts, observations (qualitative).
- **Analysis**: Frequency counts, percentages, thematic coding, triangulation.

3.1 Quantitative Findings

Student Engagement Compared to Normal Programming Classes
- Much more engaging: 39.3%
- More engaging: 53.6%
- Same: 7.1%

Enjoyment in Working with Microcontroller
- Rating 5: 35.7%
- Rating 4: 42.9%
- Rating 3: 21.4%

Logic Building Through Projects
- 84.2% agreed projects improved logic understanding.

Preferred Learning Method
- Hands-on projects: 42.9%
- Combination of AI + practical: 53.6%
- AI-only: 3.5%

AI Tool Usage vs Independent Thinking
- 92.9% had used AI tools.
- 88.8% agreed hardware activities encouraged independent thinking.

3.2 Qualitative Findings

Students' reflections highlighted the **joy of tangible coding**:
- They enjoyed *actually playing* the **Rock-Paper-Scissors game** they coded.
- The **Digital Dice** project was fun because they could shake the Micro:bit to roll dice.
- The **Footstep Tracker** connected programming to fitness and health.
- The **Alzheimer's Patient Tracker** (radio communication) showed coding as socially meaningful.
- The **Earthquake Warning Device** using the shake sensor gave students a sense of coding for safety.

Quotes:
- *"Problem-solving skills, identifying the error, learning to code."*
- *"Freedom to implement your own ideas and create unique projects."*
- *"It was fun, practical, and engaging."*

4. Discussion

Physical computing shifted programming from abstract syntax exercises to **visible, interactive, and meaningful experiences**. Students not only wrote code but also *played games, shook devices, tracked footsteps, and designed warning systems*. This tangible connection between code and real-world outcomes enhanced **computational thinking** and **learner agency**.
- **Engagement**: 92.9% found activities more engaging than traditional programming.
- **Logic & CT**: Hardware projects demanded iterative debugging, strengthening computational thinking.
- **Creativity**: Projects like the Alzheimer's Tracker illustrate programming as invention.
- **Balancing AI**: While nearly all students used AI tools, only 3.5% preferred AI-only learning. Physical computing provided the counterbalance.
- **Confidence**: Students reported increased confidence when their code produced tangible results.

5. Educational Implications

1) Programming instruction should emphasize **project-based learning**.
2) Physical computing reinforces Python concepts while making them relevant.
3) AI should be integrated responsibly, without replacing critical thinking.
4) Hardware-based learning promotes creativity, teamwork, and confidence.
5) CBSE schools can adopt Micro:bit modules to future-proof programming education.

6. Limitations

The results of this classroom action research are encouraging, but there are a number of constraints to consider:

1) Subjects: 28 students of a single CBSE affiliated school. Results may not be applicable to other schools, regions or educational systems.
2) Duration of Intervention: While the intervention was of short duration (40 days), it did allow for measuring short-term outcomes, but not for long-term retention, sustained engagement, and transfer to other programming contexts.
3) Micro:bit constraints: There was limited availability of Micro:bit devices which meant that some students were required to share Micro:bit devices to work on, thereby reducing the amount of time each student could spend working with Micro:bit devices. There were also some hardware disconnections and outdated components which impacted the flow of the learning process.
4) Self-Reported Data: Much of the quantitative information was gathered via surveys, which can lead to errors in students' self-assessment. Future studies may benefit from using observational and performance-based measures.
5) Platform Specificity: The study was conducted on only the BBC Micro:bit. It is effective, but may not work with other platforms of physical computing like arduino or raspberry pi.
6) AI Tools: Students indicated that they were able to use AI alongside hands-on learning, but the study did not systematically examine the impact of using AI to support coding and physical computing.
7) Urban School Context
8) Integration with AI Tools: Although students reported balancing AI with hands-on learning, the study did not systematically measure how AI-assisted coding influenced outcomes when combined with physical computing.

7. Future Scope

This study opens several avenues for further exploration:

1) Expansion to Rural Schools: Since the present research was conducted in an urban school with access to resources, future studies should examine the impact of physical computing in rural or resource-constrained settings. This would help determine whether similar gains in engagement and computational thinking can be achieved where infrastructure is limited.
2) Multi-School and Larger Samples: Replicating the study across multiple CBSE schools and with larger student populations would strengthen the generalizability of findings and allow for comparative analysis across socio-economic contexts.
3) Long-Term Retention Studies: Extending the intervention beyond 40 days would enable measurement of long-term retention, sustained engagement, and transfer of skills to other programming contexts.
4) Comparative Platforms: Future research could compare the BBC Micro:bit with other microcontrollers such as Arduino, Arduino Nano, NodeMCU, and Raspberry Pi. These platforms offer varying levels of complexity, connectivity, and project possibilities, and comparative studies could reveal which devices are most effective for different age groups, curricula, and learning goals.
5) Hybrid AI–Hardware Models: Systematic studies could explore how AI-assisted coding tools can be integrated with physical computing to create hybrid learning models that balance efficiency with deep problem-solving.
6) Curriculum Integration: Investigating how physical computing can be embedded into the CBSE Computer Science curriculum at scale, including assessment frameworks and teacher training, would support sustainable adoption.
7) Real-World Applications: Projects such as the Alzheimer's Patient Tracker, Footstep Tracker, and Earthquake Warning Device highlight the potential for socially meaningful coding. Future work could expand into health, safety, and environmental monitoring applications, connecting student learning with community impact.

8. Conclusion

In the age of AI-assisted coding, physical computing offers a powerful pedagogy to sustain **computational thinking, creativity, and learner agency**. By integrating Micro:bit-based projects into Python instruction, CBSE classrooms can move beyond superficial coding to foster **engaged, independent problem-solvers** prepared for real-world challenges.

Acknowledgement
The author would like to thank the students of Class XI Computer Science of the senior secondary school affiliated to CBSE, which participated in this study. They enthusiastically explored using the BBC Micro:bit and shared their reflections, allowing this research to take place. Many thanks to the Computer Science faculty for their cooperation, and to the school administration for their resources and encouragement. The author is also grateful to the constructive comments from the peers in the design and analysis phase of this classroom action research.

References

[1] Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. Proceedings of the 2012 Annual Meeting of the American Educational Research Association.
[2] Papert, S. (1980). Mindstorms: Children, computers, and powerful ideas. Basic Books.
[3] Resnick, M. (2017). Lifelong kindergarten: Cultivating creativity through projects, passion, peers, and play. MIT Press.
[4] Smith, J., Lee, A., & Kumar, R. (2024). AI-assisted coding in education: Opportunities and challenges. Computers & Education, 205, 104456.
[5] Wing, J. M. (2006). Computational thinking. Communications of the ACM, 49(3), 33–35.
[6] Yurdakök, A., & Kalelioğlu, F. (2024). The impact of physical computing activities on students' computational thinking skills. Journal of Educational Technology & Society, 27(1), 45–58.