A Preliminary Analysis of Gamification on Assembly Language Programming Education

Muhtar Çağkan Uludağlı
The Department of Computer Engineering, İzmir University of Economics, Turkey

cagkan.uludagli@ieu.edu.tr

Abstract: Assembly language is often perceived as complex and abstract, necessitating instructional approaches that enhance student engagement and comprehension. This study presents a preliminary analysis of gamification methods in assembly programming education, focusing on the challenges of teaching low-level programming concepts. A total of 18 studies incorporating gamification strategies are reviewed to synthesize insights into the pedagogical benefits and limitations of gamification in assembly education. The analysis categorizes studies by gamification techniques, course contexts, and research methodologies, identifying recurring themes in enhancing motivation, skill acquisition, and understanding of fundamental concepts. The methodology of this study involved a structured search query across multiple academic databases, including IEEE Xplore, ACM Digital Library, and ScienceDirect. The search was iteratively refined, and a snowballing approach was adopted to identify additional studies. From an initial pool of 162 studies, 47 were considered relevant, and 18 were selected for analysis. The review follows a chronological structure, highlighting key contributions and connections between studies. Research methodologies used in studies include qualitative and quantitative analyses, experimental comparisons, and practical implementations. The reviewed studies illustrate various gamification techniques, including interactive learning environments that visualize assembly operations, commercial games incorporating assembly-like programming, badge-based learning systems that promote structured progression, and game-based project assignments that introduce assembly concepts through game development. Findings indicate that gamification enhances student engagement by integrating real-time feedback, hands-on learning, and competitive elements. Projects such as Pong, Breakout, and TetrisOS demonstrate that game-based assignments can make assembly concepts more accessible. Studies on badge-based learning highlight how structured rewards encourage students to develop assembly skills at their own pace. Some studies suggest that gamification must be complemented by additional instructional strategies to maximize long-term retention. In summary, this study underscores gamification's role as a pedagogical tool for assembly programming education, showing its effectiveness in fostering student engagement and comprehension.

Keywords: Gamification, Game-Based learning, Assembly programming, Low-Level programming, Microprocessor

1. Introduction

In recent years, gamification has emerged as a powerful pedagogical tool across numerous educational fields, enhancing engagement, motivation, and retention by incorporating game-like elements into learning experiences (Dichev & Dicheva, 2017; Koivisto & Hamari, 2019). Particularly in technical and challenging subjects such as assembly programming and its usage in microprocessor architectures, gamification has shown promising potential. These subjects, integral to computer engineering curricula, are often perceived as complex due to their abstract and intricate nature (Logozar et al, 2022). As a result, students frequently struggle with grasping low-level programming concepts, understanding hardware intricacies, and applying these in real-world contexts (Kelleher & Pausch, 2005). Gamification offers a novel approach to mitigating these challenges by transforming traditional educational frameworks into more interactive and engaging experiences.

Assembly language programming is a low-level form of coding that communicates directly with a computer's hardware. Each assembly instruction corresponds closely to an operation executed by the CPU, giving the programmer fine-grained control over what the processor does. This direct control makes assembly powerful and efficient for tasks requiring hardware-level optimization, but it also means the programmer must manage many details manually—such as *register manipulation*, which involves modifying data in the CPU's fast storage units; *conditional branching*, where program flow is controlled based on logic conditions; and *memory addressing*, where exact memory locations must be specified rather than using variable names. Unlike high-level languages (such as Python or Java) that abstract away these details, assembly requires the developer to handle every operation explicitly, demanding a deep understanding of computer architecture and resulting in a steep learning curve for newcomers.

Despite the growing adoption of gamification in various technical fields, its application within assembly programming education remains less explored. While educators and researchers have recognized its potential, the unique complexities of these subjects necessitate a more targeted analysis. Assembly language requires a distinct set of cognitive skills, involving precise logic, attention to detail, and a deep understanding of hardware-software interactions. By leveraging gamification, educators aim to create more intuitive pathways for students

to engage with these concepts, potentially enhancing both learning outcomes and students' self-efficacy in tackling these demanding topics.

This analysis examines existing research on gamification in the context of assembly programming education. By synthesizing findings from studies across computer engineering and educational psychology, this analysis aims to provide insights into how gamification can be strategically implemented in the area. It covers a range of gamification techniques, including point systems, challenges, and simulation-based exercises, and analyzes their impact on student motivation, comprehension, and skill acquisition.

The structure of this paper is organized as follows: this section outlines the significance of gamification in making complex subjects like assembly language and microprocessor architecture more accessible. Methodology explains the search process, including the initial structured query and the subsequent snowballing approach to locate relevant studies. In Analysis, key findings from the identified studies are presented, categorizing different gamification methods used in technical education and examining their impact on learning outcomes. The Discussion synthesizes the benefits and limitations observed, noting areas where gamification effectively enhances engagement. Finally, the Conclusion summarizes the contributions of this work and underscores the potential of gamified learning as a valuable educational tool in assembly programming education.

2. Methodology

The research for this analysis began with a structured search query in major academic databases, such as IEEE Xplore, ACM Digital Library, and ScienceDirect, to identify relevant studies on gamification and game-based learning usage in assembly language programming education. The initial search query, structured to capture studies that integrate gamification with low-level programming education, was:

> *(("All Metadata":gamif*) OR ("All Metadata":gamified) OR ("All Metadata":gamification) OR ("All Metadata":game-based learning)) AND (("All Metadata":assembly language) OR ("All Metadata":assembly program*) OR ("All Metadata":microcontroller) OR ("All Metadata":microprocessor) OR ("All Metadata":processor)) AND (("All Metadata":education) OR ("All Metadata":learning))*

However, this query yielded a limited number of studies directly relevant to the intersection of gamification and assembly language education, likely due to the specificity of these topics. Recognizing this limitation, the query was relaxed in stages to capture a broader range of relevant studies on gamification in technical education settings, even if they did not specifically address assembly instructions.

After getting limited results from the query, a snowballing approach was adopted, whereby key studies identified through the initial search were used as seeds to locate additional relevant research. This involved reviewing references within these studies and exploring works that cited them, a recognized method for literature reviews that aim to capture a broader scope when primary search results are limited (Wohlin, 2014). This approach allowed for a more comprehensive inclusion of literature, including works on gamification's application to similar technical subjects, such as digital systems and programming fundamentals, where assembly language principles are often indirectly addressed.

Overall results of the search query on different databases and the snowballing approach yielded a total of 162 initial studies. Out of these results, 47 of them are considered related with our work to a certain degree. Detailed analysis and readings decrease the number of reviewed studies. In final, 18 studies are analyzed in this work.

3. Analysis

In this analysis, a chronological order for reviewing the studies is followed like the similar studies (Manzano-León et al, 2021). Where applicable, the connections between the studies are explicitly pointed out.

Sprunt (2005) presents a structured, gamified approach to teaching assembly language through a Pong game project that employs problem-based learning (PBL) to engage introductory students in developing a fully functional game in assembly. By requiring interaction with microcontroller hardware and implementation of features like paddle control and ball movement using polling and interrupts, the assignment promotes deep engagement with low-level programming. Developed in structured stages aligned with Bloom's taxonomy, the project moves from basic mechanics to advanced features like collision detection and scoring, encouraging students to apply, analyze, and synthesize assembly concepts such as register manipulation, conditional branching, and memory addressing. Feedback indicates that the gamified, hands-on, and competitive format

enhances motivation and supports a deep-learning approach, making assembly programming enjoyable while fostering independent learning and improving retention.

Kawash & Collier (2013) explore the use of video game creation to teach assembly language in a second-year course at the University of Calgary, where students developed games like Space Invaders, Tetris, and Pac-Man entirely in assembly to increase engagement through a "fun factor" without reducing curriculum rigor. This approach required students to address complex hardware/software interface tasks such as programming interrupt handlers, using VESA BIOS Extensions (VBE) for graphics, and developing custom keyboard and mouse drivers via direct port and memory-mapped I/O. By building interactive games, students practically learned low-level concepts including memory management, hardware interrupts, and device interface manipulation, simulating real-time programming challenges. Survey results showed that 65% of students enjoyed the project and found the gaming focus improved their understanding and engagement with assembly programming despite its difficulty.

Jamieson (2014) examines badge-based learning in a computer architecture course to enhance assembly language instruction by replacing traditional lectures and tests with a system where students earned basic, intermediate, and advanced badges through demonstrated mastery. This format enabled self-paced learning, allowing motivated students to explore advanced assembly topics while others met minimum requirements with basic badges. Key skills included CPU operations, registers, instruction sets, memory addressing, and debugging, assessed through practical deliverables and instructor interviews. Unlike one-time assessments, students could refine their work for each badge, promoting deeper learning. While grade distributions remained similar to the traditional format, the badge-based model improved engagement and supported mastery of low-level assembly concepts, particularly among high-achieving students.

Johnson et al (2016) evaluates the use of video game methodologies, particularly gamification, to enhance assembly language education by integrating games like The Foos, Lightbot, Picobot, and especially Human Resource Machine, which uses a corporate-themed puzzle format to introduce assembly-like operations such as load, jump, and conditional branching. This game provides a visual, interactive method for understanding memory operations and sequential logic, while its "corporate ladder" structure presents progressively complex puzzles that mirror the iterative and optimization-focused nature of assembly programming. With dual-metric scoring on program length and runtime efficiency, it teaches practical trade-offs in code design. The study highlights that such gamified approaches increase engagement and deepen comprehension of low-level programming by embedding assembly concepts into gameplay, thus lowering cognitive barriers and supporting active learning of foundational skills necessary for advanced programming and hardware interaction.

Black et al (2017) outlines a lab project for teaching assembly language and computer organization through developing a simplified x86 version of Breakout, where students create game elements like the ball, paddle, and blocks with collision detection. The project extends learning by having students build an 8088-based computer on a breadboard using components like LEDs, push-buttons, and an Atmega microcontroller to emulate video and keyboard interfaces. After testing the game in assembly, students load the machine code onto an EEPROM chip, turning their breadboard into a standalone gaming device with an LED matrix display and push-button controls. This hands-on, gamified approach deepens understanding of low-level programming, hardware interfacing, and real-time code execution, with student feedback showing increased engagement and comprehension of core computer organization concepts.

Black (2017) introduces TetrisOS and BreakoutOS, two projects for computer organization courses where students build basic x86 assembly versions of Tetris and Breakout on bare-metal PCs, enabling direct hardware interaction without an operating system. These projects cover memory addressing, device communication, interrupt handling, and OS fundamentals through progressive steps; in BreakoutOS, students implement video memory, game objects, keyboard input, and collision detection, while TetrisOS adds timer interrupts and nested loops for gameplay. Each concludes with creating a bootable USB for standalone execution. Students reported high engagement and found the projects made assembly language more accessible and practical through interactive, tangible outcomes.

Cass (2017) evaluated a set of games that make assembly-like programming accessible and enjoyable by using simplified virtual assembly environments focused on low-level instructions like data movement, conditional branches, and arithmetic, helping players intuitively grasp assembly principles. Each game offers distinct scenarios: Human Resource Machine uses office-themed puzzles to teach memory and register concepts; TIS-100 simulates a retro microcomputer with parallel-processing puzzles and limited instruction sets; Shenzhen I/O challenges players to design and optimize virtual electronic circuits. These immersive, puzzle-based games

promote understanding of low-level coding by encouraging players to think like assembly programmers in an engaging, interactive way.

Black & Matta (2019) details a sophomore-level computer organization project where students develop Breakout using AVR assembly on an Arduino Uno, engaging with core assembly concepts by building a functional game with an 8x8 LED matrix display and push buttons for paddle control. Without system calls, students directly manage I/O and control logic, progressing through sequential steps that introduce wiring circuits, creating variables, handling delays with loops, implementing conditional branches and collision detection, and using hardware interrupts for paddle movement. The hands-on, gamified approach provided immediate visual feedback, helping students grasp abstract assembly concepts, face real-world challenges like timing and debugging, and gain practical insight into low-level computing, with feedback showing enhanced understanding and engagement.

Dicheva et al (2020) explores the use of badges to boost engagement and motivation in a sophomore-level computer science course that includes assembly language programming, using the OneUp platform to incentivize out-of-class exercises on core topics like assembly fundamentals. After an initial non-gamified phase, badges were introduced for achievements in areas such as quizzes and exercises, leading to increased participation, particularly in voluntary assembly tasks, and improved performance on assembly-focused assessments and overall grades. While the badges effectively encouraged extrinsically motivated engagement and consistent practice essential for mastering low-level programming, the survey based on the Basic Psychological Needs Satisfaction Scale (Deci et al, 2000) showed no significant increase in intrinsic motivation. Thus, badges functioned as external motivators, helping students internalize the value of assembly tasks over time, making them a practical gamified supplement in assembly education.

Gallego-Durán et al (2021) explore a bottom-up approach to programming education through the DEZ80 course, which uses Z80 assembly language and game development challenges to address shortcomings of teaching only high-level languages that may lead to superficial understanding. A 16-hour course with first-year students showed that early exposure to low-level programming led to similar or slightly better performance in later programming assessments, indicating deeper comprehension. The gamified, challenge-based format promoted engagement and supported low-level programming as a valuable complement to traditional methods for developing computational thinking.

Llamas-Nistal et al (2022) describes a hands-on approach to teaching computer architecture using ARM assembly on Raspberry Pi, replacing simulators with real hardware to enable tangible interaction with machine-level concepts. Students begin with foundational tasks on QtARMSim to learn basic operations, instruction sets, and memory management, then transition to Raspberry Pi projects involving GPIO control, delay loops, I/O handling, interrupts, and subroutine calls. Gamified tasks like a “Simon Says” game or traffic light controller help students observe real-time code effects, while the alignment with ARM architecture and interactive environment enhances motivation and learning by connecting assembly programming to real-world applications.

Gryphon & Chung (2023) introduces Assembly Academy, an educational game that simplifies assembly language learning through programming challenges where students control a virtual robot using commands modelled after ARM assembly. By providing real-time visual feedback, error messages, and step-by-step execution, the game demystifies abstract concepts and supports learning through immediate correction and comprehension. Each level introduces a new ARM command applied in puzzles that reinforce its use, with commands like “botmove,” “botgrab,” and “botlook” mirroring fundamental operations and registers R0 to R7 used to simulate limited memory management. The game’s scoring system, based on memory efficiency and clock cycles, fosters motivation and code optimization in a competitive, gamified environment, making assembly language more approachable and engaging.

Calvo-Morata et al (2024) investigates gamification as a strategy to attract students, particularly young women, to programming and computational thinking, aiming to make STEM more accessible and engaging. The Game4Coding Erasmus+ project is introduced, emphasizing the development of CodeQuest, a video game designed within the monster-tamer genre.

The paper briefly discusses using video games to teach programming concepts, even at low levels such as assembly language. It references games like Human Resource Machine, TIS-100, and Shenzhen I/O, which are specifically designed to introduce complex, low-level programming concepts in an engaging way. These games incorporate assembly-like programming challenges where players must solve problems using a command set

that mimic assembly language instructions. From this study, we find two other studies to be analyzed, (Kawash & Collier, 2013; Cass, 2017), using snowballing method.

Dolinsky et al (2024) reviews challenges and advancements in teaching computer science fundamentals across various educational contexts. It emphasizes the incorporation of modern methods such as gamification, virtual laboratories, and artificial intelligence-based personalized learning to address gaps in student engagement and knowledge acquisition. The article highlights key topics in computing education, including computer architecture, assembly programming, and programming languages, and presents the Computing Curriculum 2020 framework as a strategic guideline. This article also references other related works providing further insights into effective gamification strategies on assembly.

Larraza-Mendiluze et al (2024) explores game-based learning to teach foundational computer architecture and assembly language, using interactive games to engage students with concepts like instruction cycles, memory addressing, and control structures. A key example is the "SUN Game," where students manually execute instructions to understand the fetch-decode-execute cycle, using a simplified register and memory system to increment variables, control loops, and perform arithmetic, closely mirroring real assembly tasks. As the course progresses, students play various games aligned with different assembly topics, applying skills like conditional branching, loops, and stack management. The study found that this approach improved motivation, retention, and both practical and theoretical understanding compared to traditional methods.

Rivera-Alvarado & Guadamuz (2024a) argues for rigorous experimentation in assembly language education, highlighting the cognitive complexity of low-level programming and the limitations of relying on qualitative feedback without formal assessment. They introduced a hands-on method inspired by board game mechanics, where students manually adjust registers and memory to simulate instruction execution. In a control experiment with five participants, students completed an assembly challenge using this method and a similar task on a computer a week later; paired t-test analysis showed no significant performance difference, though students reported greater engagement and interactivity. The study emphasizes the need for formal evaluation of teaching methods and plans to expand participant samples to refine the hands-on approach.

Rivera-Alvarado & Guadamuz (2024b) reviews the potential of video games as tools for assembly language instruction, noting the subject's complexity despite its importance in areas like embedded systems. They analyze games such as "TIS-100", "Human Resource Machine", and "Exapunks", which use pseudo-assembly for puzzle-solving but omit essential elements like stack usage, memory addressing, and system calls, limiting their effectiveness for genuine assembly skill development. While these games offer design insights, the authors propose creating an educational assembly language game and evaluating its impact through controlled experiments assessing both theoretical and practical proficiency to establish a scientific basis for game-based learning in assembly education.

Furthermore, Rivera-Alvarado (2024) explores using video game technology to enhance assembly language learning by proposing a game that teaches real assembly through controlling an avatar with programmed instructions, translating complex tasks into interactive scenarios. The study plans a structured experiment comparing this method to traditional teaching and incorporates a multimodal feedback system—visual, haptic, and auditory—to reinforce concept understanding. Aiming to fill a gap in gamified tools for assembly, the research seeks to establish a robust framework for immersive, interactive learning in technical programming education.

To summarize, the studies analyzed in this chapter (the details of which are also given in Table 1) reveal that gamification in assembly language and microprocessor education can effectively enhance student engagement and comprehension through diverse, hands-on approaches. By integrating challenge-based learning, game-based projects, and structured progression systems, these methods help demystify complex low-level programming concepts, making them more accessible and motivating for students. While the outcomes show increased skill acquisition and motivation, they also suggest that gamification strategies alone may benefit from supplementary instructional support to fully foster long-term retention and deeper understanding in technical subjects.

Table 1: Summary of the analyzed studies with their details

Study	Gamified Element	Assembly Focus	Key Findings
Sprunt (2005)	Pong game in assembly	Problem-based learning, hardware interaction	Increased motivation and deep learning
Kawash & Collier (2013)	Space Invaders, Tetris, Pac-Man in assembly	Game development with interrupts and I/O	Enhanced understanding and engagement
Jamieson (2014)	Badge-based learning	Self-paced mastery over assembly topics	Greater depth of understanding, flexible learning
Johnson et al (2016)	Human Resource Machine	Puzzle-based assembly logic	Visual understanding, iterative optimization
Black et al (2017)	Breakout game on breadboard	Bare-metal programming with x86	Hands-on learning, deepened hardware insight
Black (2017)	TetrisOS and BreakoutOS	Bare-metal x86 game projects	High engagement, step-by-step skill building
Cass (2017)	Human Resource Machine, TIS-100, Shenzhen I/O	Simplified virtual assembly	Interactive learning, low-level instruction understanding
Black & Matta (2019)	Breakout on Arduino Uno	AVR assembly with LED matrix	Immediate feedback, hardware debugging
Dicheva et al (2020)	Badge system via OneUp	Gamified out-of-class exercises	Extrinsic motivation, increased quiz scores
Gallego-Durán et al (2021)	DEZ80 course	Z80 assembly with game challenges	Improved foundational understanding
Llamas-Nistal et al (2022)	ARM on Raspberry Pi	QtARMSim + real hardware	Visualization of real-world assembly effects
Gryphon & Chung (2023)	Assembly Academy	ARM-based robot programming	Progressive feedback, real-time visualization
Calvo-Morata et al (2024)	CodeQuest	Monster-tamer game with assembly	Gender-inclusive, challenge-based learning
Dolinsky et al (2024)	Review of CS education	Gamification, AI, virtual labs	General trends supporting gamified assembly
Larraza-Mendiluze et al (2024)	SUN Game	Manual fetch-decode-execute simulation	Improved motivation and conceptual clarity
Rivera-Alvarado & Guadamuz (2024a)	Board game mechanics	Manual register/memory manipulation	Positive feedback, no significant performance change
Rivera-Alvarado & Guadamuz (2024b)	Review of game-based learning	Human Resource Machine, Exapunks, etc.	Proposes deeper, experimental educational games
Rivera-Alvarado (2024)	Proposed a video game	Real assembly gameplay with multimodal feedback	Structured evaluation plan, immersive learning

4. Discussion

This study places itself within an emerging body of research that explores gamification in the teaching of assembly language and related technical subjects in computer engineering. The use of game-based learning techniques, ranging from virtual robots to badge-based assessments, reflects a significant shift in pedagogy aimed at mitigating the challenges traditionally associated with low-level programming concepts. The studies reviewed reveal several recurring themes, methodologies, and implementations of gamification in computer science and engineering education, with a particular focus on improving student engagement, motivation, and comprehension.

The thematic gamification areas used by the analyzed articles are:

- **Interactive Learning Environments and Visualization (ILEV):** Many studies have employed interactive environments to make assembly language and microprocessor concepts more tangible. For instance, Assembly Academy and SPIMbot use virtual robots to allow students to visualize low-level operations in real-time. By controlling robots with assembly commands, students can see immediate results of their code execution, which aids in demystifying abstract operations such as memory management and register manipulation.

- **Commercial Games as Educational Tools (CGET):** Several studies integrated commercial games like Human Resource Machine, Exapunks, Shenzhen I/O and TIS-100 to expose students to simplified, assembly-like programming environments. These games typically focus on core programming constructs which mirror assembly language skills in a more accessible format. While these games lack the full depth of assembly programming, they provide valuable introductory experiences and can enhance engagement with foundational programming logic.
- **Badge-Based Learning Systems (BBLS):** Badge-based systems, as explored in multiple studies, provide a structured framework for students to progress at their own pace. These systems reward students for mastering specific tasks, thereby promoting consistent practice and reducing the pressure of high-stakes assessments. Findings from these studies suggest that badge-based approaches are especially beneficial in courses that cover complex microprocessor architectures like ARM and x86, as they allow students to solidify foundational skills before advancing.
- **Game-Based Project Assignments (GBPA):** Assignments are centred around game creation, such as those seen in TetrisOS, BreakoutOS, and a Pong game project, introduce students to core assembly concepts in a step-by-step manner. These projects employ game mechanics to gradually introduce students to complex tasks. Such projects provide a ground-up experience by engaging students directly with hardware interfaces.

The studies utilize diverse research methodologies to examine the effectiveness of gamification in technical education. Common methods include comparative experiments, such as badge-based vs. traditional grading structures, control and experimental groups using game-based projects, and user feedback collection to gauge engagement and comprehension. Some studies, such as the work with Assembly Academy (Gryphon & Chung, 2023), implement structured tutorials followed by hands-on challenges, enabling researchers to track learning progression and identify areas where students struggle.

The evaluation techniques, including statistical analysis and qualitative feedback, offer insight into both the benefits and limitations of gamification in assembly education. For instance, while several studies report increased student engagement and skill development, some findings suggest that gamification alone may not significantly enhance intrinsic motivation (Dicheva et al, 2020). Additionally, feedback from game-based projects reveals that students often find real-time feedback and tangible outcomes, such as LED matrix displays in the Arduino Arcade Machine project (Black & Matta, 2019), helpful in overcoming the cognitive challenges associated with assembly language.

The analysis shows that most of the studies are from the last 10 years, with only one study from early 2000s. There are five different games that are recurrently used in gamification purposes of assembly language education. They are usually chosen for their similarity to assembly language or the use of assembly-like commands in commercial games type, and for their ease of modification and portability in early games such as Pong or Tetris. The gamification of the assembly education mostly occurs in introductory programming, assembly programming or microprocessor architecture courses. Five different research types are employed, including practical, experimental, quantitative, qualitative or general review studies. The most used assembler and microprocessor architectures are x86 and ARM.

Table 2 below summarizes the gamification methods, assembly lecture contexts, research types, commercial games used, and microprocessor architectures or assembler information from the studies discussed in this paper. Table uses the acronyms provided in this section for gamification methods column for easy reading. Some studies are analyzed, but removed from the summary table (i.e., Calvo-Morata et al, 2024; Dolinsky, 2024), since they are the studies that introduce other works or do not apply any gamification method to be presented in the table.

Table 2: The summary table on gamification in assembly language education. "HRM" acronym denotes the game "Human Resource Machine"

#	Gamification	Course Application	Research Type	Commercial Games	Microprocessor or Assembly Architecture
(Sprunt, 2005)	CGET & GBPA	Computer Architecture	Practical	Pong	General Assembly
(Kawash & Collier, 2013)	CGET & GBPA	Computer Architecture	Qualitative	Space Invaders, Tetris, Pac-Man	x86
(Jamieson, 2014)	BBLS	Computer Architecture	Quantitative	None	Atmel, PIC, HC11, ARM
(Johnson et al, 2016)	CGET	Computer Science Courses	Qualitative	The Foos, Lightbot, HRM	General Assembly
(Black et al, 2017)	CGET & GBPA	Computer Architecture	Practical	Breakout	x86
(Black, 2017)	CGET & GBPA	Computer Architecture	Practical	Tetris, Breakout	x86
(Cass, 2017)	Assembly-like Games	None	Review	HRM, TIS-100, Shenzhen I/O	General Assembly
(Black & Matta, 2019)	CGET & GBPA	Computer Architecture	Practical	Breakout	AVR Assembly, Arduino
(Dicheva et al, 2020)	BBLS	Introductory Programming	Experimental	None	MIPS
(Gallego-Durán et al, 2021)	GBPA	Assembly Programming	Experimental	None	Z80
(Llamas-Nistal et al, 2022)	GBPA	Computer Architecture	Practical	None	ARM
(Gryphon & Chung, 2023)	ILEV	Assembly Programming	Practical	None	ARM
(Larraza-Mendiluze et al, 2024)	ILEV	Computer Architecture	Experimental	None	General Assembly
(Rivera-Alvarado & Guadamuz, 2024a)	GBPA	Assembly Programming	Experimental	None	None
(Rivera-Alvarado & Guadamuz, 2024b)	Assembly-like Games	Assembly Programming	Review	TIS-100, HRM, Exapunks	General Assembly
(Rivera-Alvarado, 2024)	GBPA	None	Review	None	General Assembly

5. Conclusion

This analysis examined the use of gamification in assembly programming education, noting both potential benefits and limitations. The literature suggests that incorporating game-like elements (such as badge-based systems, interactive simulations, and competitive challenges) can improve student motivation and engagement, and may also support practical skill development in this traditionally complex subject. Using these techniques could make learning assembly more approachable for students by helping to lower some of the initial barriers associated with low-level programming.

However, while gamification shows promise, it also presents unique challenges. Issues such as cognitive overload, variability in student response, and the need for content-specific adaptation underscore the importance of refining gamified approaches. The impact of gamification on long-term retention and deep conceptual understanding is currently inconclusive, suggesting the need for further longitudinal studies. Addressing these gaps will require a more tailored approach to gamification, one that aligns with the distinct cognitive demands of assembly programming.

In conclusion, while gamification alone is unlikely to resolve all educational challenges in assembly programming, it represents a valuable tool in the pedagogical toolkit for technical education. Continued research into adaptive, subject-specific gamification frameworks may yield approaches that more effectively integrate game-based elements with deep technical content. This ongoing exploration will be essential to developing educational practices that both motivate students and facilitate meaningful learning in assembly programming courses.

Ethics declaration: The author declares that the present study did not require ethical approval by an institutional review board or ethics committee.

AI declaration: The author used generative AI tools to assist in improving the clarity and structure of the manuscript. All content was critically reviewed and fact-checked by the author to ensure accuracy and originality, and the final manuscript reflects the author's own interpretations and contributions.

References

Black, M. (2017). TetrisOS and BreakoutOS: Assembly language projects for computer organization. *Proceedings of the 2017 ACM Conference on Innovation and Technology in Computer Science Education*, 88–89. https://doi.org/10.1145/3059009.3072976

Black, M., Brady, C., & Goulski, T. (2017). *Breakout from x86 assembly to a breadboard: A lab for introductory computer organization*. *33*(1), 131–138.

Black, M., & Matta, J. (2019). Teaching AVR assembly by building an arduino arcade machine. *Proceedings of the 2019 ACM Conference on Innovation and Technology in Computer Science Education*, 233–234. https://doi.org/10.1145/3304221.3325549

Calvo-Morata, A., Humble, N., Mozelius, P., Pechuel, R., & Fernández-Manjón, B. (2024). Games for coding to attract new students to STEM. *2024 IEEE Global Engineering Education Conference (EDUCON)*, 01–08. https://doi.org/10.1109/EDUCON60312.2024.10578626

Cass, S. (2017). Some assembly (language) required—Three games that make low-level coding fun [Resources_Geek Life]. *IEEE Spectrum*, *54*(5), 19–20. https://doi.org/10.1109/MSPEC.2017.7906890

Deci, E. L., Ryan, R. M., Gagné, M., Leone, D. R., Usunov, J., & Kornazheva, B. P. (2001). Need satisfaction, motivation, and well-being in the work organizations of a former eastern bloc country: A cross-cultural study of self-determination. *Personality and Social Psychology Bulletin*, *27*(8), 930–942. https://doi.org/10.1177/0146167201278002

Dichev, C., & Dicheva, D. (2017). Gamifying education: What is known, what is believed and what remains uncertain: A critical review. *International Journal of Educational Technology in Higher Education*, *14*(1), 9. https://doi.org/10.1186/s41239-017-0042-5

Dicheva, D., Caldwell, R., & Guy, B. (2020). Do badges increase student engagement and motivation? *Proceedings of the 21st Annual Conference on Information Technology Education*, 81–86. https://doi.org/10.1145/3368308.3415393

Dolinsky, M. (2024). Trends in the development of basic computer education at universities. *Buletin Ilmiah Sarjana Teknik Elektro*, *5*(4), 584–591. https://doi.org/10.12928/biste.v5i4.9704

Gallego-Durán, F. J., Satorre-Cuerda, R., Compañ-Rosique, P., Villagrá-Arnedo, C. J., Molina-Carmona, R., & Llorens-Largo, F. (2021). A low-level approach to improve programming learning. *Universal Access in the Information Society*, *20*, 479–493.

Gryphon, K., & Chung, H. (2023). Assembly academy: Using video games and virtual robots to teach assembly programming. *2023 IEEE International Conference on Advanced Learning Technologies (ICALT)*, 108–110. https://doi.org/10.1109/ICALT58122.2023.00037

Jamieson, P. (2014). Does badge-based learning buck the grading curve? An educational experiment in computer architecture. *Proceedings of the International Conference on Frontiers in Education: Computer Science and Computer Engineering (FECS)*, 1.

Johnson, C., McGill, M., Bouchard, D., Bradshaw, M. K., Bucheli, V. A., Merkle, L. D., Scott, M. J., Sweedyk, Z., Velázquez-Iturbide, J. Á., Xiao, Z., & Zhang, M. (2016). Game development for computer science education. *Proceedings of the 2016 ITiCSE Working Group Reports*, 23–44. https://doi.org/10.1145/3024906.3024908

Kawash, J., & Collier, R. (2013). Using video game development to engage undergraduate students of assembly language programming. *Proceedings of the 14th Annual ACM SIGITE Conference on Information Technology Education*, 71–76.

Kelleher, C., & Pausch, R. (2005). Lowering the barriers to programming: A taxonomy of programming environments and languages for novice programmers. *Acm Computing Surveys*, *37*(2), 83–137. https://doi.org/10.1145/1089733.1089734

Koivisto, J., & Hamari, J. (2019). The rise of motivational information systems: A review of gamification research. *International Journal of Information Management*, *45*, 191–210. https://doi.org/10.1016/j.ijinfomgt.2018.10.013

Larraza-Mendiluze, E., Arbelaitz Gallego, O., Arregi Uriarte, O., Ignacio Martín Aramburu, J., & Francisco Lukas Mugika, J. (2024). Basic concepts of computer architecture through games and game development. *IEEE Revista Iberoamericana de Tecnologias Del Aprendizaje*, *19*, 195–204. https://doi.org/10.1109/RITA.2024.3458863

Llamas-Nistal, M., Fernández-Iglesias, M. J., Santos-Gago, J. M., Anido-Rifón, L. E., Mikic-Fonte, F. A., Liz-Domínguez, M., & Pacheco-Lorenzo, M. R. (2022). Towards a new approach to the Computer Architecture lab. *2022 Congreso de Tecnología, Aprendizaje y Enseñanza de La Electrónica (XV Technologies Applied to Electronics Teaching Conference)*, 1–4. https://doi.org/10.1109/TAEE54169.2022.9840747

Logozar, R., Horvatic, M., Sumiga, I., & Mikac, M. (2022). Challenges in teaching assembly language programming – desired prerequisites vs. Students' initial knowledge. *2022 IEEE Global Engineering Education Conference (EDUCON)*, 1689–1698. https://doi.org/10.1109/EDUCON52537.2022.9766737

Manzano-León, A., Camacho-Lazarraga, P., Guerrero, M. A., Guerrero-Puerta, L., Aguilar-Parra, J. M., Trigueros, R., & Alias, A. (2021). Between level up and game over: A systematic literature review of gamification in education. *Sustainability*, *13*(4), Article 2247. https://doi.org/10.3390/su13042247

Rivera-Alvarado, E. (2024). Video game technologies applied for teaching assembly language programming. *Proceedings of the 26th International Conference on Multimodal Interaction*, 617–621. https://doi.org/10.1145/3678957.3688622

Rivera-Alvarado, E., & Guadamuz, S. (2024a). A case for the importance of formal experimentation in teaching programming: A brief study with assembly language. In J. Choudrie, E. Tuba, T. Perumal, & A. Joshi (Eds.), *ICT for intelligent systems* (pp. 11–20). Springer Nature Singapore.

Rivera-Alvarado, E., & Guadamuz, S. (2024b). A review of the use of video games for purposes besides entertainment: A case for a novel approach for teaching assembly language. In A. K. Nagar, D. S. Jat, D. K. Mishra, & A. Joshi (Eds.), *Intelligent sustainable systems* (pp. 241–250). Springer Nature Singapore.

Sprunt, B. (2005). The pedagogical advantages of using the pong video game for problem-based learning in an introductory assembly-language programming course. *COMPUTERS IN EDUCATION JOURNAL*, *15*(4), 9.

Wohlin, C. (2014). Guidelines for snowballing in systematic literature studies and a replication in software engineering. *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering*. https://doi.org/10.1145/2601248.2601268