**Arduino Scoreboard Using Algorithms**

|  |  |
| --- | --- |
| Anh Vo  Computer Science  Auburn University at  Mongomery  Montgomery, United State avo@aum.edu | Ryan Yang  Computer Science  Auburn University at Montgomery, United States  ryang1@aum.edu |

This project presents the design and implementation of an Arduino-powered smart scoreboard system that utilizes algorithmic logic and a user-friendly Qt-based graphical interface. The system responds to external inputs, updates scores in real time using LED displays, and leverages computational techniques including greedy algorithms, dynamic programming, and recursive functions to manage score changes efficiently. The project aims to provide a reliable, low-cost, and responsive solution for real-time score management in sports and games. Development was carried out using Arduino hardware, C++ for embedded logic, and Qt for front-end application design. This paper outlines the full design methodology, the integration of algorithmic logic, and the implementation results..

***Keywords—component, formatting, style, styling, insert***

# **(**key words**)**

*I. INTRODUCTION (HEADING 1)*

In modern sports and competitive gaming environments, realtime score tracking is essential for ensuring audience engagement and fair play. Traditional manual systems are prone to human error and delay. The purpose of this project is to design and build a smart scoreboard system powered by Arduino technology that automates score updates using optimized algorithms. This system improves performance, accuracy, and ease of use, and it integrates a Qt-based desktop interface to allow users to update scores interactively.

|  |
| --- |
| XXX-X-XXXX-XXXX-X/XX/$XX.00 ©20XX IEEE |

The core problem addressed by this project is the latency and inaccuracy associated with manual scorekeeping. By embedding algorithmic logic in the score tracking process, the system can manage updates quickly and reliably, ensuring real-time feedback with minimal hardware resource consumption. The goals of this project are: (1) to implement a functional and efficient scoreboard system using Arduino, (2) to integrate algorithmic strategies to handle real-time score

# II. LITERATURE REVIEW

# A. Remote Controlled Embedded System Based Scoreboard Design with Mobile Program

In their 2022 study, Boyacı and Tümbek developed a scoreboard system utilizing an Arduino Uno microcontroller, a dot matrix display, a real-time clock (RTC) module, and Bluetooth communication for mobile control. The system allowed users to update scores and display messages via a mobile application, demonstrating a practical application of embedded systems in sports settings. While effective in integrating hardware and mobile interfaces, the study did not delve into algorithmic optimizations for performance enhancement.

# B. Understanding Algorithm Awareness in Digital

SystemsJones, M. (2014). CHI '14 Extended Abstracts on Human Factors in Computing Systems. This work explores how users interact with and perceive algorithm-driven systems. The paper inspired the integration of algorithms in our scoreboard, not only to improve technical performance but also to study their unnoticed impact on real-time processes, similar to how algorithms curate digital feeds*.*

# III. METHODOLOGY

The development of the Arduino scoreboard system was completed through four main stages: hardware integration, software development, algorithm implementation, and GUI construction using Qt.

# A. Hardware Setup

The hardware includes an Arduino Uno, LED displays for score output, push buttons for input, and optional Bluetooth modules for remote control. The circuit was first designed on a breadboard and later transitioned to a permanent form.

# B. Software Development with Qt

Qt was chosen as the development platform for the GUI due to its flexibility and support for serial communication. The desktop application allows users to send score changes to the Arduino. The GUI provides buttons, labels, and status indicators to represent real-time changes.

# C. Algorithm Integration

To optimize performance and manage score transitions, three computational paradigms were implemented:

* Greedy Algorithms: Used for immediate score updates while minimizing LED changes.
* Recursive Functions: Helped in resetting or calculating score thresholds.
* Dynamic Programming: Cached prior score states to improve efficiency during updates.
* **Dijkstra’s Algorithm**: Implemented to compute the shortest path between score states in certain game modes where transitions between complex states are required. This ensured optimal decision-making in score management with minimal hardware delay.

# IV. CONCLUSION

The Arduino-based smart scoreboard successfully demonstrates how embedded systems, when combined with efficient algorithms and GUI development, can provide an affordable, scalable, and accurate scorekeeping solution. The use of Qt for interface design made the project more user friendly and enhanced user interaction. This project helped the team develop practical skills in hardware interfacing, C++ programming, algorithm design, and GUI development—core competencies for computer science students.

Future improvements include integrating sensor-based inputs, wireless connectivity enhancements, and further optimization of the score update algorithms. This hands-on experience significantly contributed to our growth in embedded systems and software engineering.

# REFERENCES

[1] [1] Boyacı, O., & Tumbek, M. (2022). Remote Controlled Embedded System Based Scoreboard Design with Mobile

Program. In Versatile Multidisciplinary Engineering Research (pp. 61-76). SRA Academic Publishing.

[3] [2] Jones, M. (2014). CHI '14 Extended Abstracts on Human Factors in Computing Systems. Association for Computing Machinery.

**IEEE conference templates contain guidance text for composing and formatting conference papers. Please**

**ensure that all template text is removed from your conference paper prior to submission to the**

**conference. Failure to remove template text from your paper may result in your paper not being published.**