

A Project Report on

Implementing Checkers Using IoT

Submitted in partial fulfilment of award of

BACHELOR OF TECHNOLOGY

Degree in

COMPUTER SCIENCE AND ENGINEERING

By

KRATIKA -1900820100062
DHAIRYA HANS-1900820100037
DHRUV RASTOGI -1900820100038
JATIN -1900820100056

Session: 2019-2023

Kanchan
Assistant Professor

SUPERVISOR



**Department of Computer Science & Engineering
Moradabad Institute of Technology
Moradabad (U.P.)**

MAY 2023

CERTIFICATE

Certified that the Project Report entitled “**Implementing Checkers Using IoT**” submitted by **Kratika (1900820100062), Dhairya Hans (1900820100037), Dhruv Rastogi (1900820100038) and Jatin (1900820100056)** is their own work and has been carried out under my supervision. It is recommended that the candidates may now be evaluated for their project work by the University.

Date:

Kanchan
Assistant Professor

ABSTRACT

The proposed research focuses on the development of an IoT-based version of the popular board game, Checkers. In this project, we aim to enhance the traditional Checkers game experience for children by incorporating a program on the board that can predict the best moves for the computer to win or lose the game. The game is built using an IoT-based approach, which allows the players to play the game on a physical board with embedded sensors. The sensors detect the movement of pieces on the board and transmit the data to the program running on the Arduino Uno microcontroller board. The system uses magnetic hall sensors to detect the movement of pieces and to identify the current position of the pieces on the board.

The research has involved the development of a custom PCB (Printed Circuit Board) to interface with the hall sensors and Arduino Uno board. The program running on the Arduino Uno board is responsible for processing the sensor data, predicting the best moves for the computer, and updating the position of the pieces on the board. The development of this system has required the integration of several technologies, including circuit design, programming, and IoT.

The system has been tested and evaluated to ensure that it works accurately and effectively. The results indicate that the IoT-based Checkers game is a promising approach to enhance the traditional board game experience for children. This research has significant implications for the use of IoT-based technologies in board games and can be extended to other board games. The proposed system can be used in schools and homes to promote problem-solving skills and to foster an interest in technology among children.

ACKNOWLEDGEMENT

It brings us immense pleasure to present this project report on **Implementing Checkers Using IoT**, which is a crucial component of the curriculum for the Bachelor's In Technology degree in computer science and engineering at Moradabad Institute of Technology, affiliated to Dr. APJ Abdul Kalam Technical University, Lucknow. The journey of working on this project was nothing short of extraordinary, and it provided us with an opportunity to acquire an array of new skills. However, the completion of this project would not have been possible without the invaluable support and assistance of several individuals.

We would like to take this opportunity to express our sincere gratitude to all those who have helped us along the way. Our heartfelt thanks go out to Dr. Manish Gupta, Dr. Neelaksh Sheel, Mr. Vikas Mittal, Mr. Anurag Malik, and Mr. Himanshu Agarwal for their unwavering support during the project's development and the preparation of this report. We also extend our deepest appreciation to all the teachers, particularly our HOD Dr. Manish Gupta, and our Director Dr. Rohit Garg for their guidance, continuous supervision, and

for providing us with the necessary information and resources to complete the project successfully.

Last but not least, we would like to acknowledge the invaluable support we received from our parents and friends. Their unwavering encouragement and cooperation helped us overcome the project's challenges and enabled us to complete it successfully. We are grateful for their unwavering support and encouragement, and we extend our heartfelt thanks to them.

KRATIKA (1900820100062)

DHAIRYA HANS (1900820100037)

DHRUV RASTOGI (1900820100038)

JATIN (1900820100056)

TABLE OF CONTENTS

ABSTRACT.....	iii
TABLE OF CONTENT.....	vii
LIST OF FIGURES.....	x
CHAPTER 1: INTRODUCTION.....	12-22
1.1 Introduction.....	13
1.2 Problem Statements.....	14
1.3 Purpose.....	16
1.4 Industrial/Society Benefits.....	19
1.5 Related Work.....	20
1.6 Methods.....	21
CHAPTER 2: TECHNOLOGY SPECIFICATION.....	23-34
2.1 Internet of Things.....	23
2.2 Arduino Uno.....	24
2.3 Hall Sensors.....	27
2.4 Shift Register.....	28
2.5 RGB LEDs.....	29
2.6 Arduino IDE.....	31
2.7 Outer Glass.....	32
2.8 Wooden Box.....	32
2.9 Programming Language.....	33
CHAPTER 3: LITERATURE REVIEW.....	35-39
3.1 Checkers game Software Version	35
3.2 Project Proposed Technique.....	36
CHAPTER 4: MODULE WISE DESCRIPTION.....	40-45
4.1 Working Principle.....	40

4.2 Module wise Division.....	41
4.2.1 Piece Moves.....	41
4.2.2 Piece Capture.....	41
4.2.3 Piece Becomes King.....	42
4.2.4 King Movement.....	42
4.2.5 Winning Condition.....	43
4.2.6 Draw Condition.....	44
4.2.7 Time Condition.....	44
CHAPTER 5: ALGORITHM DESIGN.....	46-62
5.1 Game Algorithm.....	46
5.1.1 Game State Matrix.....	47
5.1.2 LEDs Structure Matrix.....	49
5.1.3 Piece Movement Algorithm.....	51
5.1.4 Possible Moves Algorithm.....	52
5.1.5 Piece Selection Algorithm.....	53
5.1.6 Display Moves Algorithm.....	54
5.1.7 B&W Piece Tracking Algorithm.....	57
5.1.8 King Making Algorithm.....	58
5.1.9 King Movement Algorithm.....	59
5.1.10 King Cut Algorithm.....	61
CHAPTER 6: DESIGN AND IMPLEMENTAION.....	63-66
6.1 Flow Diagrams.....	63
6.2 Dimension Used.....	64
6.3 Working Board.....	65
CHAPTER 7: FEATURES.....	67-68
CHAPTER 8: LIMITATIONS.....	69-70
CHAPTER 9: RESULTS.....	71-76
CHAPTER 10: CONCLUSION.....	77-78
CHAPTER 11: FUTURE WORK.....	79-80

REFERENCE.....	81
APPENDIX A: RESEARCH PAPER.....	82

LIST OF FIGURES

Fig.No.	Title	Page No.
1.1	Internet of Thing.....	12
2.1	Arduino Uno.....	24
2.2	Arduino uno With Circuit.....	25
2.3	Hall sensor A3144.....	27
2.4	74HC165 shift register.....	28
2.5	74HC165 shift register Pins.....	29
2.6	WS2812B RGB LED strip.....	30
2.7	Arduino IDE.....	31
2.8	Outer Glass.....	32
2.9	C++ Language.....	33
3.1	Electronic Circuit.....	37
3.2	Game Board Design.....	37
3.3	Main Circuit Diagram.....	38
4.1	Main Circuit.....	40
5.1	Matrix Representation.....	48
5.2	Piece value representation.....	49
5.2	Vertical Moves Show.....	54
5.3	Horizontal Moves Show.....	55
5.4	King Movements.....	60
6.1	Flow Chart.....	63
9.1	Showing movies.....	71
9.2	Turn change to player 2.....	71
9.3	Showing moves of player 2.....	72
9.4	Showing cut move by player 1.....	72
9.5	Showing cut piece of player 2.....	73
9.6	Double cut move.....	73
9.7	Show king making move.....	74

9.8	Show king piece.....	74
9.9	Show king moves.....	75
9.10	Show king 4 moves.....	75
9.11	Player 1 winning move.....	76
9.12	Player 1 wins.....	76

CHAPTER-1

INTRODUCTION

The Internet of Things (IoT) is a system of interconnected physical devices that can interact and share data without the need for human involvement. Because IoT allows us to gather information from all kinds of mediums, such as humans, animals, automobiles, and household appliances, it has been explicitly characterized as an "Infrastructure of the Information Society." By integrating electronic hardware within any physical device that may be given an IP address, including detectors, application, and networking gear. The project's objective is to develop a checkers Board game using IoT on a physical board that can predict and show the possible moves of pieces with the help of LEDs.



Fig.1.1- Internet of Things

1.1 Introduction

Board games have been a popular pastime among people of all ages for centuries, providing entertainment and mental stimulation as in Fig.1.1. In recent years, advancements in technology have opened up new possibilities for board games, enabling developers to create more interactive and engaging experiences for players. One such advancement is the integration of the Internet of Things (IoT) into board games, which allows for real-time tracking of moves and interactions between players.

The game of checkers, also known as draughts, has been a favourite board game for many generations. It is a two-player game played on an 8 by 8 board with 64 squares, and each player has 12 pieces. The objective of the game is to capture all of the opponent's pieces or to block their pieces in such a way that they cannot make any more moves. The game is easy to learn but can be challenging to master, making it a popular choice for players of all ages.

We present a project on Checkers Using IoT, which aims to enhance the traditional game of checkers by integrating IoT technology. The project utilizes an 8 by 8 wooden board with 64 grids, on which the game is played. The board is connected to an Arduino Uno, which is a microcontroller board with multiple input and output pins. The board also contains 32 hall sensors, which are used to detect the movement of the pieces on the board.

The main objective of this project is to develop an IoT-based checkers game that can predict the best moves for the computer to win or lose the game. This is accomplished by programming the Arduino Uno to detect when a piece is lifted from the board, and then using a computer algorithm to analyse the board position and suggest the optimal move for the computer player. By incorporating IoT technology into the game of checkers, we aim to provide a more engaging and interactive experience for players.

The rest of this report is organized as follows. We provide a detailed description of the hardware and software components used in the project. This is followed by a section on the system architecture, which describes how the various components are connected and how they interact with each other. We then present the algorithm used for predicting the best moves, along with a description of the implementation details. Finally, we conclude the report with a discussion of the results and the potential future directions for this project.

The Checkers Using IoT project presented in this report is an innovative approach to enhancing the traditional game of checkers by incorporating IoT technology. The project utilizes an Arduino Mega microcontroller board, 32 hall sensors, and a computer algorithm to predict the best moves for the computer player. By providing a more engaging and interactive experience for players, we hope to inspire a new generation of board game enthusiasts and promote the use of IoT technology in board games.

1.2 Problem Statement

The game of checkers has been popular for centuries and has evolved over time. With the advent of new technologies, it is now possible to create a more interactive and engaging version of the game using IoT. However, there are several challenges that need to be addressed before this can be achieved. The following are some of the main problems that the proposed research aims to solve:

- 1. Lack of interactivity:** Traditional board games, including checkers, are relatively static and lack the interactivity that modern-day children are used to. They often require multiple players to be physically present, and there is little scope for automation or customization.
- 2. Limited learning potential:** Although checkers can be an excellent tool for developing critical thinking and problem-solving skills, the traditional game has

limited potential in terms of learning. It can become monotonous and repetitive after a while, and children may lose interest.

3. **Accessibility issues:** Some children may not have access to traditional checkers boards or may find it challenging to set up the game correctly. This could lead to frustration and a lack of engagement.
4. **High-level programming skills:** Developing an IoT-based checkers game from scratch requires a high level of programming skills, and there may be a shortage of people with this expertise in some areas.

To address these problems, the proposed research aims to create an IoT-based version of the popular board game checkers. This will involve the use of sensors and microcontrollers to automate the game and provide a more interactive and engaging experience for children. The following are some of the specific objectives of the research:

- To develop a system that allows checkers to be played on a digital board with IoT technology.
- To automate the game using sensors and microcontrollers so that moves can be tracked and predicted.
- To customize the game to make it more interactive and engaging for children, with features such as RGB light on each grid, error showing, timer.
- To evaluate the effectiveness of the IoT-based checkers game in terms of its ability to foster critical thinking and problem-solving skills in children.

By addressing these challenges, the proposed research aims to create a more accessible, engaging, and effective version of the popular board game checkers. The project will

leverage IoT technology to enhance the learning potential of the game and make it more interactive and engaging for children.

1.3 Purpose

- **To develop and test a complete algorithm for playing checkers:**

This project serves as an opportunity to design, implement, and test a fully functioning algorithm that can play checkers at a high level. By working on this project, developers can improve their programming and problem-solving skills while also gaining a deeper understanding of the game of checkers.

- **To showcase the capabilities of machine learning:**

Checkers is an ideal game for testing the capabilities of machine learning algorithms. This project can demonstrate how machine learning techniques can be used to develop intelligent agents that can learn from experience and make informed decisions based on game states.

- **To improve the quality of game-playing algorithms:**

Developing algorithms that can play games at a high level is a complex task that requires a deep understanding of the game mechanics and strategies. This project can contribute to improving the quality of game-playing algorithms by providing a platform for developers to test and refine their algorithms in a competitive environment.

- **To provide a fun and engaging application of machine learning:**

Checkers is a classic game that is enjoyed by people of all ages. By developing an algorithm that can play the game at a high level, this project can provide a fun and engaging application of machine learning that can be enjoyed by anyone with an interest in the field.

- **To Improves problem solving techniques:**

This project aims to improve the problem-solving techniques of players through the implementation of the King Cut Algorithm. Checker is a game that requires players to think critically and strategically in order to outmaneuver their opponents. By using the King Cut Algorithm, players can develop their ability to identify potential threats to their king and find ways to protect it. This involves analyzing the board and considering various moves that can be made by both the player and their opponent. As players become more familiar with the algorithm and how it works, they can apply similar problem-solving techniques to other areas of their lives, such as in academics, career, and personal relationships.

- **Improve logic and reasoning skills:**

Playing checker requires a player to think ahead and consider all possible moves and their consequences. This process involves analysing different positions, evaluating the strengths and weaknesses of pieces, and anticipating the opponent's moves. By practicing checker, players can enhance their logical and reasoning abilities, as they learn to think critically and strategically.

When players are faced with complex checker problems, they must break them down into smaller parts and use deductive reasoning to arrive at a solution. This approach helps to improve logical thinking and promotes problem-solving skills, which can be beneficial in various areas of life.

Moreover, playing checker helps to develop analytical thinking skills, as players need to evaluate different options and their outcomes before making a move. This process requires concentration and attention to detail, which can increase a player's mental capacity and enhance their ability to focus on complex tasks.

Through regular practice and gameplay, players can also develop their creativity and intuition, as they learn to consider unusual and unexpected moves. This kind of thinking promotes flexible and adaptive thinking, which can be beneficial in various fields.

Overall, playing checker is an excellent way to improve logic and reasoning skills, as it requires players to use their mental faculties in a comprehensive and challenging way. These skills are transferable to other areas of life and can be beneficial for personal and professional growth.

- **Increase patience and persistence:**

The purpose of the project in regard to players is to increase their patience and persistence in playing the game of checker. Checker requires a great deal of patience and persistence, as players must carefully consider each move and anticipate their opponent's responses. By working on this project, players will develop a deeper understanding of the game mechanics and strategy, which will in turn help them to make more informed and successful moves. This process of trial and error, as well as the need to analyze and re-evaluate one's choices, can be frustrating at times, but it also teaches players the value of perseverance and determination. Overall, the project can help players to develop important life skills that can be applied in a variety of situations beyond the game of checker.

- **Improves decision Making skills:**

One of the primary purposes of this project is to improve decision-making skills in players. Decision-making is an essential skill that involves selecting the best course of action from a range of available options. In checker, players need to make quick and accurate decisions based on their opponent's moves and potential threats.

By playing checker and implementing the King Cut Algorithm, players can improve their decision-making skills. The algorithm requires players to carefully analyse the board and determine if their king is in danger of being captured by their opponent's piece. This process involves evaluating different options and selecting the best one to protect their king.

Moreover, as players progress through the game, they encounter more complex situations that require them to make more difficult decisions. This helps them develop their ability to analyze complex problems and identify multiple potential solutions. These decision-making skills can be transferred to other areas of life and can benefit individuals in both personal and professional settings.

1.4 Industry/Society Benefitted

The development of Checkers Using IoT offers several benefits to both the industry and society. Firstly, it offers a new and innovative way of playing the popular game of checkers that can attract a younger generation to the game. By incorporating IoT technology, this project offers a unique and interactive way to play the game that can foster an interest in technology and problem-solving skills in children.

Secondly, the use of IoT technology in Checkers Using IoT has practical applications in various industries. The development of IoT-based board games can be extended to other areas, such as manufacturing and logistics. The sensors used in this project to detect the

movement of the pieces can be used in industries to track the movement of products or machines, ensuring optimal efficiency, and reducing the risk of errors.

Additionally, the development of Checkers Using IoT can have social benefits as well. It offers a new and interactive way for individuals to engage in recreational activities, fostering social connections and providing a means for relaxation and entertainment. The project can also be used in educational settings, promoting the development of critical thinking and problem-solving skills among students.

The development of Checkers Using IoT offers benefits to both industry and society, providing a new and innovative way to play board games while also having practical applications in various industries and promoting social and educational development.

1.5 Related Work

Checkers is a popular board game that has been around for centuries. With the advancement of technology, there have been numerous attempts to digitize the game and incorporate new features. In recent years, researchers have focused on developing Checkers using IoT, allowing players to engage in a more interactive and immersive gaming experience. Here are some related works that have been done in the field of Checkers using IoT.

1. IoT-Enabled Checkers Game: A Smart Way of Learning:

This study focused on developing a Checkers game that utilizes IoT technology to enhance the learning experience of children. The game was designed to be interactive and engaging, with the IoT sensors providing real-time feedback to the players. The game also utilized a machine learning algorithm to predict the best possible moves, making it an excellent tool for improving problem-solving skills.

2. Design of a Smart Checkers Board using IoT:

This research aimed to create a smart Checkers board that utilizes IoT sensors to monitor the game and provide feedback to the players. The board was designed using an Arduino microcontroller and various sensors that detect the position of the pieces and movements on the board. The board also had a display that shows the players the current state of the game and provides tips on the best moves.

3. Smart Checkers Game with IoT:

This study focused on developing a smart Checkers game using IoT technology that allows two players to play the game remotely. The game utilized a camera to capture the movements of the pieces on the board, and the data was sent over the internet to the remote player. The game also had a chat feature that allowed players to communicate with each other during the game.

4. Checkers with Artificial Intelligence:

This research focused on developing a Checkers game with artificial intelligence that can predict the best possible moves. The game was designed to be played on a computer and utilized machine learning algorithms to analyse previous games and make predictions based on the current state of the game. The game also had a feature that allowed players to see the predicted moves, making it an excellent tool for improving problem-solving skills.

These related works have contributed significantly to the field of Checkers using IoT. The research has opened new possibilities for creating interactive and immersive gaming experiences, while also providing a valuable tool for improving problem-solving skills.

1.6 Methods

The proposed methods for developing the Checkers game using IoT include the integration of various hardware and software components. To achieve the desired outcome, the following methods were employed:

- 1. Circuit Design and Assembly:** The first step in building the Checkers game using IoT was to design and assemble the necessary circuits. The circuit was designed to fit an 8x8 board with 64 grids and 32 sensors using hall sensors and an Arduino Uno.
- 2. Programming:** The Checkers game required programming to function effectively. The programming language used was C++ and the Arduino Integrated Development Environment (IDE) was used to write and upload the code.
- 3. Sensor Calibration:** After designing and assembling the circuit, the next step was to calibrate the sensors. This involved ensuring that each sensor was properly connected and configured to provide accurate readings when a checker piece is lifted.
- 4. Predictive Modelling:** One of the core features of the Checkers game using IoT was the ability to predict moves. This was achieved using a combination of predictive modelling and algorithms in c++. The program was designed to predict the best moves for the player to win or lose the game.

By employing the above methods, the Checkers game using IoT was successfully developed. The use of IoT technology and advanced programming techniques will pave the way for future developments in the field of board games and will benefit society as a whole.

CHAPTER-2

TECHNOLOGY SPECIFICATION

2.1 Internet of Things

The Internet of Things (IoT) is a technology that connects everyday devices to the internet and allows them to communicate with each other. These devices can range from simple sensors to complex machines and are equipped with various sensors, such as temperature, humidity, and motion sensors, to collect data. The data collected by these devices is then transmitted to a central location, where it can be analysed to gain insights into patterns and trends.

In this project, we have used IoT technology to create a more interactive version of the classic board game checkers. The board is equipped with sensors, and each game piece has a magnet attached to it. The sensors are placed under each square on the board, and when a game piece is moved, the sensor detects the magnet and sends a signal to an Arduino Uno microcontroller. The microcontroller then processes the signal and determines the new position of the game piece. This data is then sent to a computer program that predicts the best move for the computer to make, thus making the game more challenging and engaging for players.

The use of IoT technology in this project has enabled us to create a more interactive and engaging game that fosters the interest of children in technology and problem-solving skills. It also provides a more seamless and accurate way to track the movements of game pieces on the board, eliminating the need for manual tracking or potential human errors. The use of IoT technology has also allowed us to collect data on how players interact with

the game, which can be used to improve the game's design and user experience in the future.

The communication protocols, networking, and connectivity utilized by web-enabled devices are largely determined by the specific IoT applications being deployed. With the help of artificial intelligence (AI) and machine learning, IoT can simplify data collection processes, making them more flexible and streamlined. The implementation of IoT technology not only enables people to live and work more efficiently but also empowers them to take complete control of their lives. In the business context, IoT plays a crucial role as it offers companies real-time insights into their systems, including the performance of machines, logistics and supply chain operations. The integration of IoT allows companies to automate processes, reduce labor costs, minimize waste, improve service delivery, and provide transparent customer transactions. Therefore, IoT technology is an essential part of everyday life, and its importance will only continue to grow as more businesses realize its potential to keep them competitive.

2.2 Arduino Uno

Arduino Uno is an open-source microcontroller board that uses the ATmega328P microcontroller as in Fig.2.1. It is widely used in various projects that require automation or control. Arduino Uno board provides several input and output pins, which can be programmed using the Arduino IDE. These pins can be used to control LEDs, motors, sensors, and other electronic components. The board also has built-in features such as analog-to-digital converters, timers, and communication interfaces as in Fig.2.2.



Fig.2.1- Arduino Uno

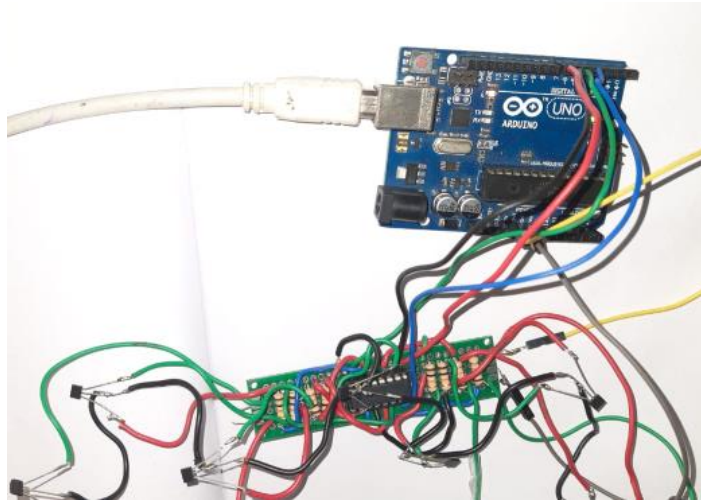


Fig.2.2- Arduino Uno with circuit

In this project, we have used Arduino Uno to interface the hall sensors with the board. Each square of the checkers board has a hall sensor attached to it, which detects the presence or absence of a checkers piece on the square. The sensor data is sent to the Arduino Uno board, which processes the data and sends it to the computer using the serial communication protocol. The board is programmed using the Arduino IDE, which is a software development environment that allows the users to write and upload the code to the board.

The Arduino Uno board is also responsible for controlling the LED lights on the board. The board is connected to the LED lights through the output pins, which can be programmed to turn on or off the lights. In our project, the LED lights are used to indicate the moves made by the computer or the player. When a move is made, the board recognizes it through the hall sensors and the LED lights indicate the move by turning on or off at the corresponding squares.

The Arduino Uno board is a versatile and essential component of this project. It provides the necessary hardware and software support to interface the hall sensors with the computer and control the LED lights on the board. The board's flexibility and easy-to-use

interface make it an ideal choice for a wide range of projects that require automation or control.

Arduino boards consist of various components, each with specific functions. The following are the components and their functionalities:

- a) Reset Button - This button restarts any loaded code on the Arduino board.
- b) AREF - Stands for "Analog Reference" and sets an external reference voltage.
- c) Ground Pin - There are multiple ground pins on the board, and they all function similarly.
- d) Analog Pins - These pins read signals from analog sensors and convert them to digital.
- e) Digital Input/Output - Pins 0-13 can be used for digital input or output.
- f) PWM - These pins can simulate analog output.
- g) USB Connection - This connection is used to power up and upload sketches to the Arduino.
- h) TX/RX - These are the transmit and receive data indication LEDs.
- i) ATmega Microcontroller - This microcontroller chip stores the programs.
- j) Power LED Indicator - This LED lights up when the board is connected to a power source.
- k) Voltage Regulator - This regulates the voltage going into the Arduino board.
- l) DC Power Barrel Jack - This powers the Arduino with a power supply.
- m) 3.3V Pin - This pin supplies 3.3 volts of power.
- n) 5V Pin - This pin supplies 5 volts of power.

2.3 Hall Sensor

Hall sensors are solid-state electronic devices that detect changes in the magnetic field. They are composed of a thin slab of semiconductor material, such as gallium arsenide, with electrical contacts attached to either end. When a magnetic field is applied perpendicular to the sensor, the electrical resistance of the semiconductor material changes. This change in resistance can be measured and used as an indication of the strength of the magnetic field.

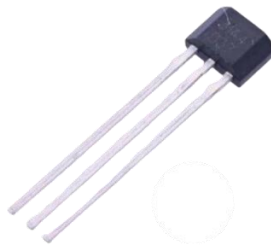


Fig.2.3- Hall sensor A3144

In the Checkers Using IoT project, we use a Hall sensor A3144 to detect the position of the checker pieces on the board. The sensor is placed under the checkerboard and connected to the Arduino Uno, which reads the sensor's output. The Hall sensor is used in conjunction with a magnet placed inside each checkers piece. When a piece is moved, the magnet's position changes relative to the sensor, and the sensor output changes. This change in the sensor output is detected by the Arduino, which then determines the position of the piece on the board.

The Hall sensor as in Fig.2.3 is an essential component in this project, as it allows for real-time tracking of the checker pieces' positions without the need for physical contact. This is important as it reduces the chances of damaging the board or pieces and provides accurate tracking of the pieces' positions. Additionally, the Hall sensor's small size makes it easy to integrate into the project without taking up too much space or affecting the board's aesthetics. Overall, the use of the Hall sensor A3144 in this project enhances the accuracy and efficiency of the system and makes it a viable solution for tracking checkers piece movements.

2.4 Shift Register

The 74HC165 is an 8-bit serial-in, parallel-out shift registers integrated circuit (IC) that is widely used in electronic projects. The shift register is a digital circuit that is used to store and transfer data serially. The 74HC165 shift register is a popular choice for many microcontroller projects because it can significantly increase the number of inputs or outputs available to a microcontroller without using many I/O pins.

To increase the output of LED's, users can connect multiple registers together. Other driver chips with "595" or "596" in their part numbers can be searched for, such as the STP16C596, which can drive up to 16 LED's and includes built-in constant current sources to eliminate the need for series resistors. The 74HC595 as in Fig.2.4 is a shift register that operates on the Serial IN Parallel OUT protocol. It accepts data in a serial manner from the microcontroller and then transmits this data through parallel pins.

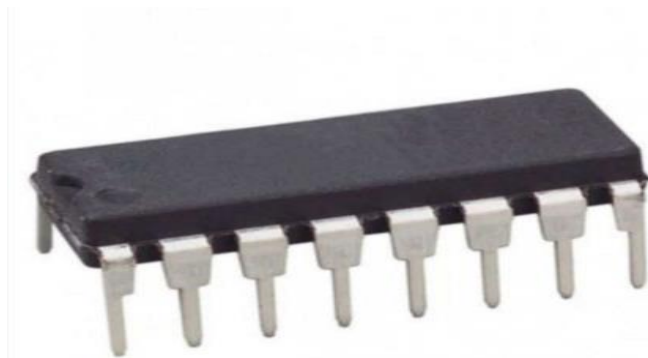


Fig.2.4- 74HC165 shift register.

In the context of our Checkers Using IoT project, we used the 74HC165 shift registers to increase the number of inputs that the Arduino Uno could handle. We used two 74HC165 shift registers, each connected to eight Hall sensors, to provide a total of 16 inputs to the Arduino Uno. When a checker piece is lifted from the board, the Hall sensor detects this change in magnetic field and sends a signal to the 74HC165 shift register, which then converts the signal from serial to parallel and sends it to the Arduino Uno. The Arduino Uno then processes the input and sends the move information to the IoT platform.

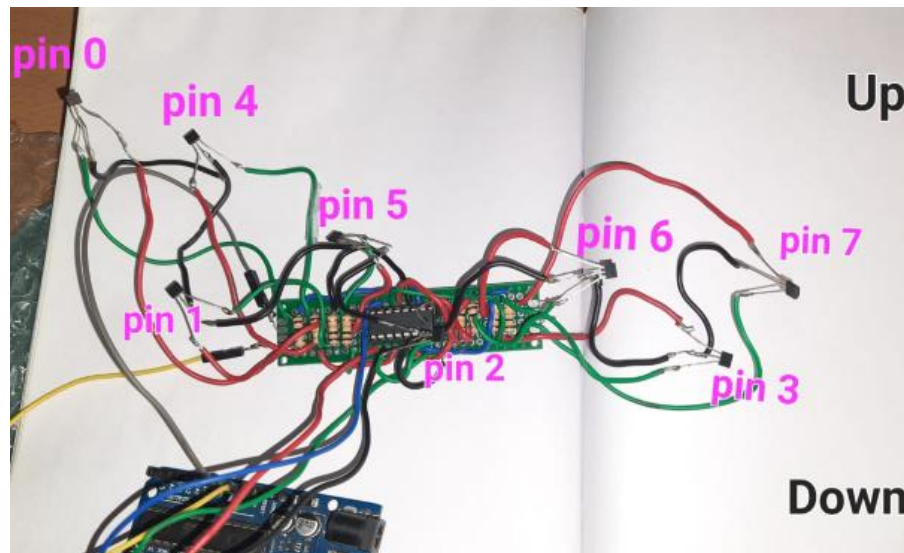


Fig.2.5- 74HC165 shift register Pins.

The 74HC165 shift registers provide a convenient and efficient way to increase the input capabilities of the Arduino Uno, which is limited to a small number of I/O pins as in Fig.2.5. By using the shift registers, we were able to greatly increase the number of Hall sensors we could use, which made the checkerboard more responsive and accurate. This allowed for a better user experience when playing checkers using our IoT system. Overall, the 74HC165 shift registers played a critical role in our project by expanding the number of inputs that the Arduino Uno could handle, which allowed us to build a more effective IoT solution for checkers.

2.5 RGB LEDs

LED stands for light emitting diode. LED lighting products produce light up to 90% more efficiently than incandescent light bulbs. An electrical current pass through a microchip, which illuminates the tiny light sources we call LEDs and the result is visible light. To prevent performance issues, the heat LEDs produce is absorbed into a heat sink.

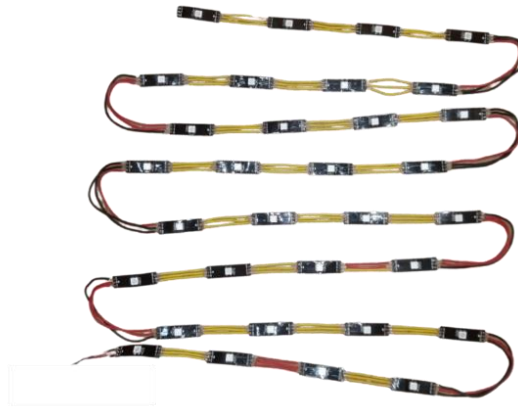


Fig.2.6- WS2812B RGB LED strip.

WS2812B as in Fig.2.6 is a popular type of addressable RGB LED strips. Each LED in the strip has a built-in controller that allows it to be controlled individually. These strips are widely used in various projects, including those that require dynamic and programmable lighting effects[2]. In this project, we used WS2812B LED strips to create a visual representation of the movements on the checker's board. Each LED in the strip represents a cell on the board, and its colour changes according to the state of the cell.

To control the WS2812B LED strips, we used an Arduino Uno and the FastLED library. The FastLED library is a high-performance library for controlling addressable RGB LED strips like the WS2812B. It provides a simple and flexible API for creating various lighting effects and animations. In our project, we used FastLED to update the colour of the LED strips in real-time, according to the movements on the checker's board.

The WS2812B LED strips are daisy-chained together, with the data line of each LED connected to the data line of the next LED. The first LED in the chain is connected to the Arduino Uno through a level shifter circuit, which converts the 5V logic of the Arduino to the 3.3V logic of the WS2812B LED strips. The level shifter circuit also provides isolation between the Arduino and the LED strips, protecting the Arduino from any voltage spikes or noise on the LED strips.

2.6 Arduino IDE

Arduino Integrated Development Environment (IDE) is an open-source software application used for programming Arduino boards. It provides a comprehensive set of tools for writing, compiling, and uploading code to the board. The IDE supports several programming languages, including C, C++, and a variant of C++ that is specific to the Arduino platform. The software can be downloaded for free from the Arduino website and is compatible with Windows, macOS, and Linux operating systems.

In this project, we used Arduino IDE as in Fig.2.7 to write the firmware code for the Arduino Uno board. The IDE provides a user-friendly interface for writing and uploading code to the board. We used the C++ programming language to write the firmware code for our project. The IDE includes a code editor, a serial monitor for debugging, and a library manager for installing and managing third-party libraries.

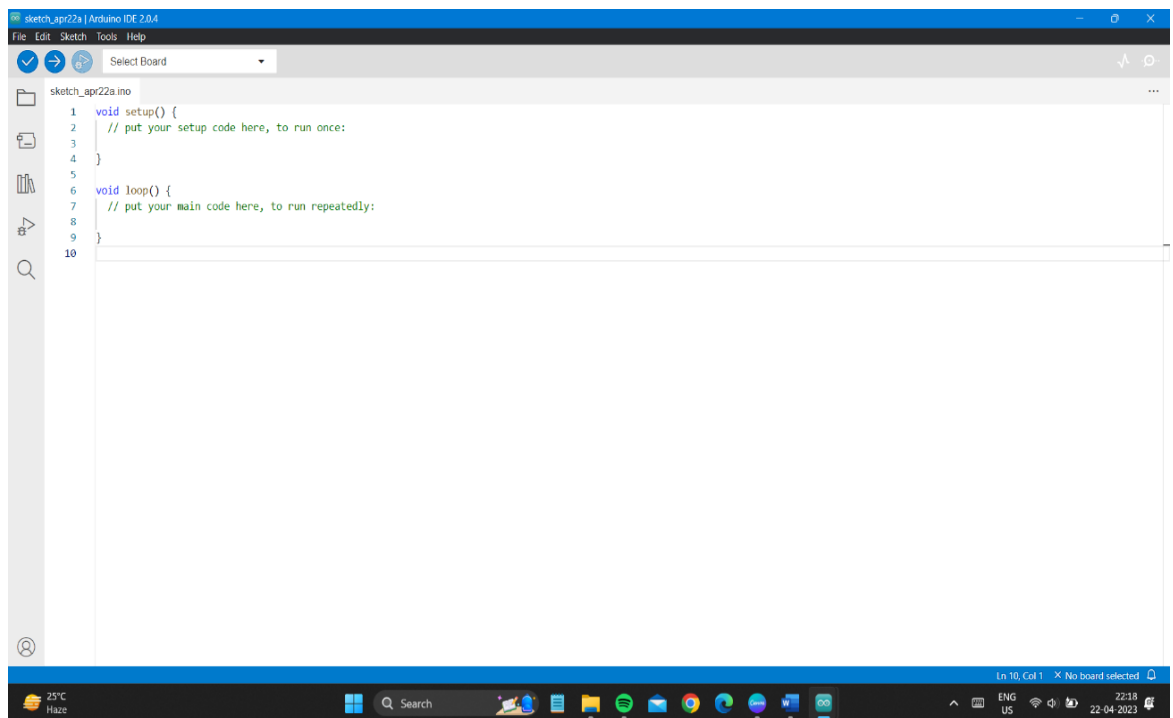


Fig.2.7- Arduino IDE

We used the Arduino IDE to write the code for interfacing the hall sensors, shift registers, and WS2812B RGB LED strips with the Arduino Uno board. The IDE provides a set of libraries that make it easy to interface with various sensors and devices. We used the

Adafruit NeoPixel library to control the RGB LED strips and the ShiftIn library to interface with the shift registers. The IDE also provides a set of example codes for different sensors and devices, which can be easily modified and integrated into the project code.

2.7 Glass

To use it as the Checkerboard, we have decided to utilize Glass material. Our plan is to exhibit the Glass as the topmost part of our Checker display. Glass is an amorphous solid material that is often transparent as in Fig.2.8 and non-crystalline. It has a wide range of practical, technological, and ornamental uses in various items like windowpanes, tableware, and optics.

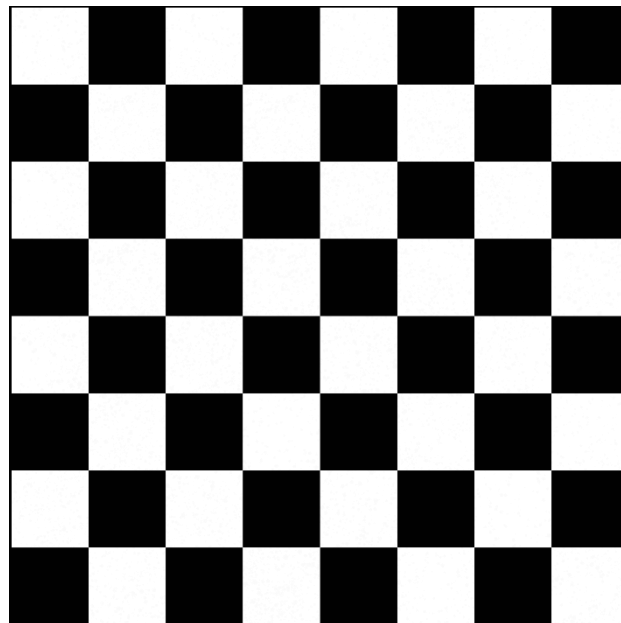


Fig.2.8- Outer Glass

2.8 Wooden Box

We have decided to incorporate a wooden box as the primary component of our Checker project. This box will serve as a container to hold all the electronic components of the project, such as wires, sensors, and the Arduino Uno. It will be attached to the Glass

display, which will serve as the Checkerboard. We have opted for a high-quality wooden material for the construction of the box, to ensure its durability and sturdiness.

2.9 Programming Language

C++ is a high-level programming language that is widely used for developing complex applications. It is an extension of the C programming language and provides additional features such as object-oriented programming, templates, and exception handling. C++ has a powerful set of libraries that make it easy to implement complex algorithms and data structures.

In this project, C++ as in Fig.2.9 is used for programming the Arduino board to interface with the various components used in the Checkers game. The Arduino board is programmed using the Arduino IDE, which provides a user-friendly interface for writing and uploading code to the board.



Fig.2.9- C++ Language

C++ is used to program the logic of the game, including the rules for moving pieces and determining the winner. The program also includes code to communicate with the hall sensors and shift registers used in the game board to detect the position of the pieces and update the LED strips accordingly.

C++ is a powerful language that allows for efficient programming of hardware, making it an ideal choice for programming microcontrollers like the Arduino. It is also widely used in the field of game development, making it a natural choice for programming the logic of the Checkers game. Overall, the use of C++ in this project enables efficient and effective implementation of the game logic and hardware interface.

According to its creator Bjarne Stroustrup, the C++ programming language consists of two primary elements: a direct mapping of hardware features primarily based on the C subset, and zero-overhead abstractions built on top of these mappings. Stroustrup characterizes C++ as "a lightweight abstraction programming language designed for creating and utilizing elegant and efficient abstractions," and emphasizes that "providing both hardware access and abstraction is at the core of C++. The language's efficiency in doing so is what sets it apart from other programming languages.

CHAPTER-3

LITERATURE REVIEW

3.1 Checkers Game Software Version

- 1. Minimax algorithm:** This is a popular algorithm used in game development that helps the computer determine the best move to make based on the current state of the game board[1]. It works by simulating all possible moves and predicting the outcome of each move several moves into the future. The move that leads to the best outcome is then selected.
- 2. Alpha-Beta pruning:** This is an optimization technique that can be used alongside the minimax algorithm. It works by pruning branches of the game tree that are unlikely to lead to a better outcome, thereby reducing the number of simulations required and improving the overall efficiency of the algorithm.
- 3. Game tree search:** This involves constructing a tree that represents all possible moves and outcomes of a game. The computer can then search this tree to determine the best move to make based on the current state of the game board.
- 4. Machine learning:** Machine learning techniques can be used to train the computer to play checkers based on a large dataset of games. The computer can learn from its mistakes and improve its gameplay over time.

5. **Heuristics:** Heuristics are rules of thumb that can be used to guide the computer's decision-making process. For example, the computer can be programmed to prioritize moves that capture the opponent's pieces or moves that lead to a more centralized position on the board.
6. **Monte Carlo Tree Search (MCTS):** This is a simulation-based search algorithm that is widely used in game development. MCTS works by randomly simulating games from the current position and selecting the move that leads to the highest win rate[5]. The algorithm can be combined with heuristics to improve its performance.

3.2 Checkers Game Software Version

1. **Microcontroller Programming:** The hardware implementation of the checkers game would require a microcontroller to handle the game logic and user input/output. The programming of the microcontroller would involve techniques such as interrupt handling, digital I/O, and serial communication.
2. **Electronic Circuit Design:** The hardware implementation would require designing electronic circuits to interface the microcontroller with the game board, buttons, LEDs, and other components. Techniques such as PCB design as in Fig.3.1, component selection, and circuit analysis would be used in this process.

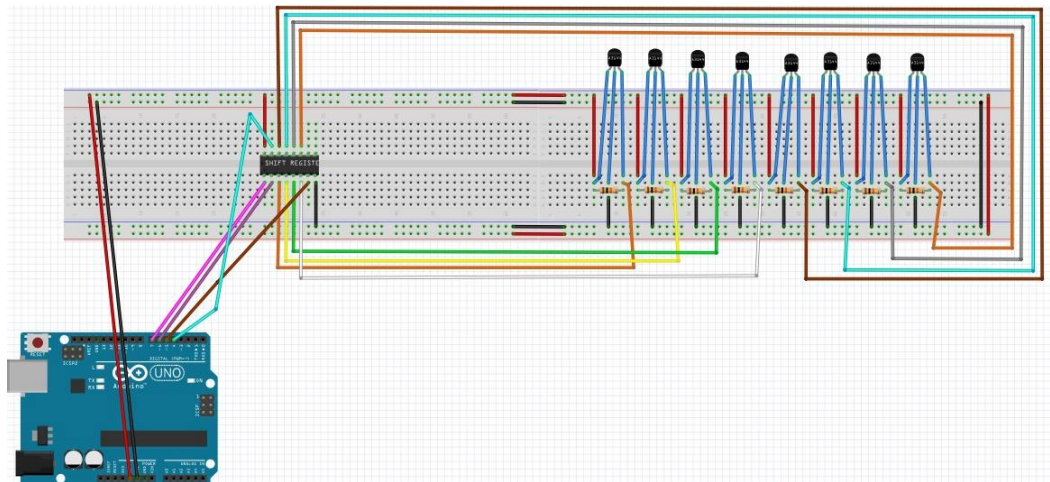


Fig.3.1- Electronic Circuit

3. **Game Board Design:** The game board design would involve techniques such as 3D modelling, prototyping, and fabrication. The board would need to be designed to accommodate the required number of game pieces and allow for easy placement and movement of the pieces as in Fig.3.2.

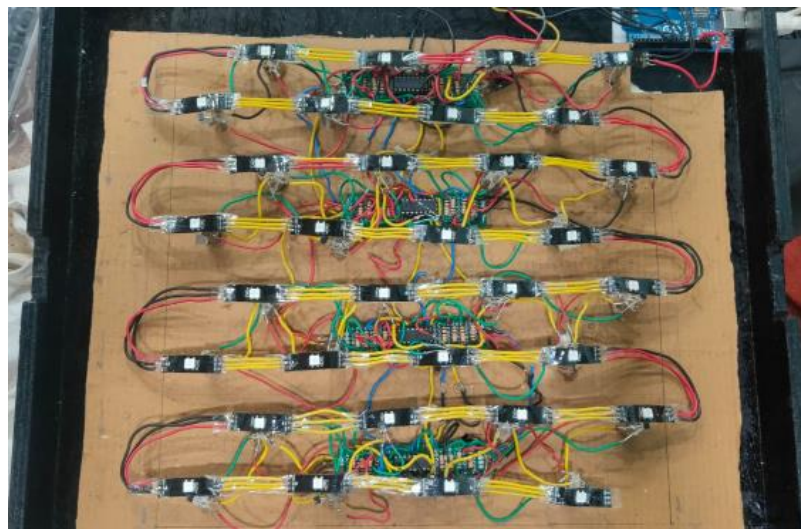


Fig.3.2- Game Board Design

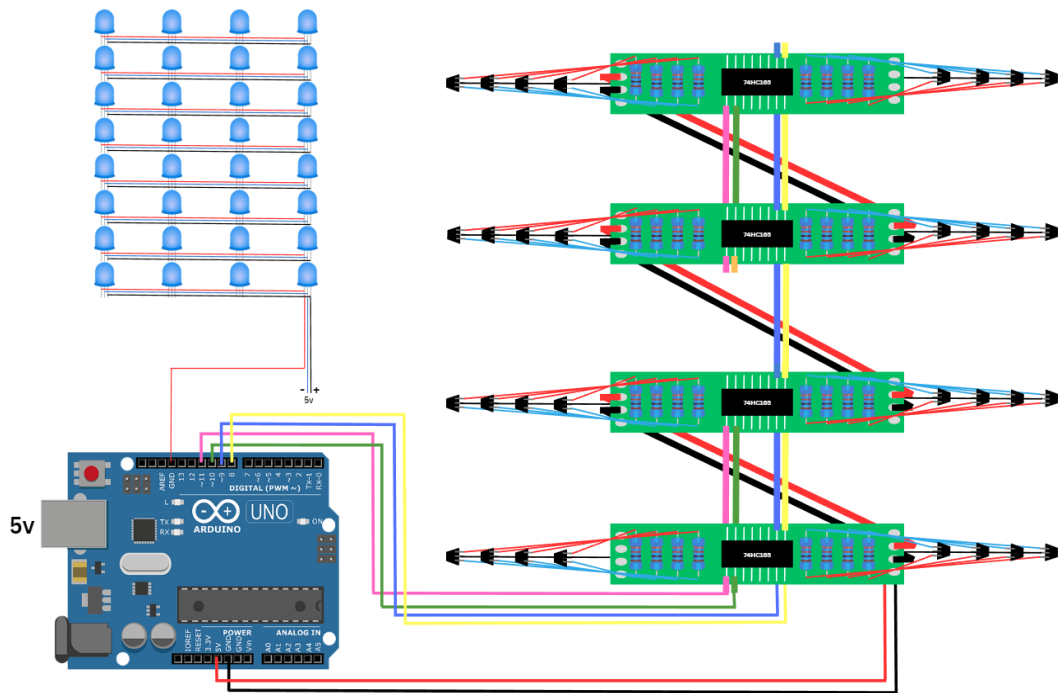


Fig.3.3- Main Circuit Diagram

4. **Testing and Debugging:** As with any hardware implementation, testing and debugging would be a critical step to ensure the game functions as intended. Techniques such as automated testing, firmware debugging, and system integration testing would be used to identify and address any issues.
5. **Sensor-based Movement:** Using sensors, such as IR sensors or Hall effect sensors, to detect the movement of pieces on the board and update the game state accordingly.
6. **FPGA Implementation:** Implementing the game on an FPGA can provide faster and more efficient processing of game logic and moves.
7. **Computer Vision:** Implementing computer vision algorithms to detect the position of pieces on the board and updating the game state, accordingly, eliminating the need for sensors or manual input.

8. **Game Analysis:** Providing game analysis features such as move suggestions and game statistics, helping players improve their game and analyse their opponents.
9. **Auto-Reset:** Automatically resetting the game board after each game or move, saving time and making the game more efficient.

CHAPTER-4

MODULE WISE DESCRIPTION

4.1 Working Principle

Checkers is a strategy board game that is played on a square board consisting of 64 squares, with alternating colours. The game is played by two players, with each player controlling a set of 12 pieces, which are typically coloured black and white, and placed on opposite ends of the board. The aim of the game is to capture all of the opponent's pieces, or to block them so that they cannot make any more moves.

Each player can only move one piece at a time, and they can only move diagonally on the board. The pieces can move forward or backward, but they cannot move sideways. A piece can only move to an empty square, and if there is an opponent's piece in the diagonal square next to it, it can capture that piece by jumping over it [2]. A player can also make multiple jumps in a single turn if the opportunity arises.

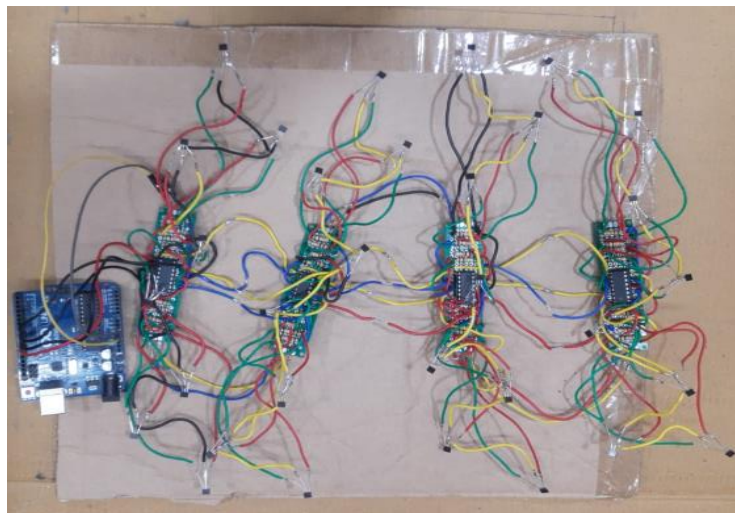


Fig.4.1- Main Circuit

The game is won by the player who either captures all of the opponent's pieces, or who blocks them so that they cannot make any more moves. A draw can also occur if both players are unable to make any more moves or if they have same number of pieces.

The game of checkers is based on simple principles of strategy, such as controlling the centre of the board, creating multiple threats, and using the strength of multiple pieces to attack weaker ones. The game requires players to think ahead and anticipate their opponent's moves in order to gain an advantage.

The working principle of checkers is based on the strategic movement of the pieces on the board and the capture of opponent's pieces to win the game. The game requires skill, planning, and a good understanding of the rules in order to be successful.

4.2 Module Wise Division

4.2.1 Piece Move

In the game of checkers, each player has a set of pieces that are typically colored black and red. The board consists of 64 squares arranged in an 8x8 grid, with each player starting with 12 pieces placed on the dark-colored squares of the three rows closest to them.

The pieces in checkers can only move diagonally, and each piece is limited to moving only one square at a time, except in certain circumstances. If a piece reaches the last row on the opponent's side of the board, it becomes a "king" and is allowed to move both forward and backward along the diagonal.

In order to capture an opponent's piece, a player must move one of their own pieces diagonally so that it lands on an adjacent square occupied by an opponent's piece. The player then removes the opponent's piece from the board and replaces it with their own piece. If a player has the opportunity to capture an opponent's piece, they are required to do so.

4.2.2 Piece Capture

In the game of checkers, a piece can capture an opponent's piece by "jumping" over it. To perform a jump, the capturing piece must be one square diagonally adjacent to the opponent's piece, and the square immediately beyond the opponent's piece must be vacant. The capturing piece then moves into the vacant square, and the opponent's piece is removed from the board. A piece that has made a capture may continue to make additional jumps in the same turn, if possible, as long as the rules for jumping are followed. When a piece reaches the last row on the opposite side of the board, it is "crowned" and becomes a king.

4.2.3 Piece Becomes King

In checkers, when a player's piece reaches the farthest row on the opposite side of the board, it is promoted to a "king." This is a significant advancement in the game as the king has more power and mobility than a regular piece. The promotion of a piece to a king is an important strategic move in the game and can greatly impact the outcome of the game. In checkers, when a player's piece reaches the farthest row on the opposite side of the board, it is promoted to a "king." This is a significant advancement in the game as the king has more power and mobility than a regular piece. The promotion of a piece to a king is an important strategic move in the game and can greatly impact the outcome of the game.

When a piece reaches the opposite side of the board, it is crowned as a king by placing another piece of the same colour on top of it. The king is recognized by having two pieces stacked on top of each other instead of just one. The player must immediately announce the promotion of their piece to a king, and it can be moved on the next turn.

Once a piece has been crowned a king, it cannot be demoted back to a regular piece. This means that the king can be used for the rest of the game and its added abilities can be used strategically to gain an advantage over the opponent. A player who has multiple kings on the board has a significant advantage, as they can be used to dominate the board and capture the opponent's pieces with ease. Overall, the promotion of a piece to a king is a critical aspect of the game of checkers and can greatly impact the outcome of the game.

4.2.4 King Movements

In Checkers, a king piece is formed by a regular piece reaching the last row on the opponent's side of the board. Once a piece becomes a king, it gains additional abilities and can move in any direction along the diagonals. This makes the king more valuable than a regular piece as it can move freely in any direction, giving the player more options to outmanoeuvre the opponent.

The king can move in a straight line along the diagonals just like a regular piece, but now it can also move backward. It can move forward and backward any number of squares along the diagonal in a single turn. This means that a king can move freely in any direction along the diagonals, which gives it more power than a regular piece. Moreover, a king can also jump over any opposing piece that is in its path, either forwards or backward. This ability to move backward and jump over any opposing piece in any direction, makes the king a very important and powerful piece in the game of Checkers.

A king is a piece in Checkers that gains additional abilities once it reaches the last row on the opponent's side of the board. The king can move in any direction along the diagonals, both forward and backward, and can jump over any opposing piece in its path. The ability to move freely in any direction and jump over pieces makes the king a valuable and powerful piece in the game, giving the player more options to outmanoeuvre the opponent.

4.2.5 Winner Condition

The winner condition in checkers is when one player captures all the pieces of the opponent or blocks their opponent's moves in a way that they can't make any more moves. When a player captures all of their opponent's pieces, the game is over, and that player is declared the winner. Capturing pieces in checkers is done by jumping over an opponent's piece diagonally. A player can jump over multiple pieces in a single turn if they are in the same diagonal line. If a player has the opportunity to capture an opponent's piece, they must take it; otherwise, the move is illegal.

In some cases, the game can end in a draw if neither player can capture all of their opponent's pieces, or they are unable to move. A draw can also occur if both players agree to end the game in a tie. In tournament play, a draw can be declared if both players have

made the same move three times in a row, or if a player has only a king left on the board, and the other player has no way of capturing it. Overall, the winner condition in checkers requires a player to use strategic planning and tactical moves to outmanoeuvre their opponent and gain control of the board.

4.2.6 Draw Condition

In a game of checkers, a draw occurs when neither player can win the game. There are several ways in which a game can end in a draw. One way is if both players have only a few pieces left on the board, and they are unable to create a forced capture or a sequence of moves that leads to a capture. In this case, the game is declared a draw.

Another way in which a game of checkers can end in a draw is if the same position is repeated three times during the game. This means that the same board position occurs three times, and the same player is to move in each of these positions. If this happens, the game is declared a draw.

Finally, a game of checkers can also end in a draw if both players agree to it. This can occur if both players feel that neither of them has a realistic chance of winning the game, or if they simply want to end the game without a winner or a loser.

4.2.7 Time Condition

In the game of checkers, players must make strategic moves to capture their opponent's pieces and control the board. However, in some cases, a player may attempt to repeatedly make the same moves in an effort to stall the game or create an advantage. To prevent this from happening, a time condition can be introduced.

We have implemented a time condition where the game lasts for a duration of 3 to 5 minutes. Once the time limit is reached, the player with more active pieces on the board is declared the winner. However, in the event that both players have an equal number of pieces on the board at the end of the time limit, the game is declared a draw. We initially

tested the time condition for 3 minutes, but we plan to increase the duration up to 5 minutes to provide players with even more challenge and excitement.

The timer condition adds an element of pressure and urgency to the game, forcing players to think more quickly and make strategic moves in a timely manner. It also adds a sense of fairness to the game, ensuring that both players have an equal opportunity to make moves and control the board. In tournaments and competitive play, the time condition is often used to ensure that games do not run on for too long and that both players have an equal chance to win. Overall, the time condition adds an interesting and challenging element to the game of checkers.

CHAPTER-5

ALGORITHM DESIGN

5.1 Game Algorithm

The base algorithm for the checkerboard game using IoT is a crucial aspect of the project. The algorithm ensures that the game is played according to the rules and is fair to both players. The algorithm is designed to provide a seamless and enjoyable experience for the players.

The Checkers game starts with the pieces placed at their default positions on the board. At the beginning of the game, the first turn is given to the player with the white pieces. The game will then show all the possible pieces that the player can move. This player can choose to move any of these highlighted pieces on the board.

Once the player decides to move a piece, the game will highlight all the possible moves that the selected piece can make. If any opponent's pieces can be cut by a move, then only those moves will be highlighted [3]. If not, all the moves will be highlighted. Once the player makes a move, the game detects whether any of the opponent's pieces have been cut.

If the player's move cuts an opponent's piece, the cut piece has to be removed from the board, the game will not continue until the cut piece is removed from the board. The game will then check whether there are any more moves with the same piece that can cut

opponent's pieces. If there are, the player will be prompted to select another move. If not, the turn will automatically shift to the opponent's turn.

The game has a timer installed that tracks the remaining time of each player to play their turn. The timer starts counting down as soon as the turn changes, and it shifts to the next player's time once the next player begins their turn. If the timer of any player reaches zero, the game ends, and the player with more pieces on the board is declared the winner. In the case of an equal number of pieces on the board, the game ends in a draw.

The game algorithm ensures that the game is played smoothly, and all the rules of checkers are followed. It also ensures that the game is fair to both players. The algorithm is designed to provide a seamless and enjoyable experience for the players. The algorithm also ensures that the game is portable due to the use of a power bank. It also helps beginners to learn the game and build logical and complex thinking skills.

It is important to note that the Checkers game algorithm is designed to be user-friendly and easy to follow. It enables players to learn the game quickly while also providing them with an engaging and challenging gameplay experience. Additionally, the use of a timer adds a sense of urgency to the game and helps to keep the gameplay fast paced and exciting.

In the future, the Checkers game algorithm could be enhanced by incorporating more advanced features such as AI vs player matches, remote play, and multiple colour LED systems. These features would make the game even more engaging and appealing to players of all skill levels. With the advancement of technology and the increasing popularity of IoT-based games, the scope for further innovation and development in the field of Checkers using IoT is vast.

5.1.1 Game State Matrix

The 'startup' matrix is a 2-D array that stores the game layout of a particular game. This matrix is used to represent the current state of the game, and it is updated each time a player makes a move. The matrix has a size of 8x8, which means it has 64 elements. In the matrix, the Black pieces are represented by the value '2', and the White pieces are represented by the value '3'. The Blank or currently empty white squares are represented by the value '0'. The Black squares are represented by the value '-1'. It is important to note that the game will only be played on White squares, which means that the Black squares will not be used during the game.

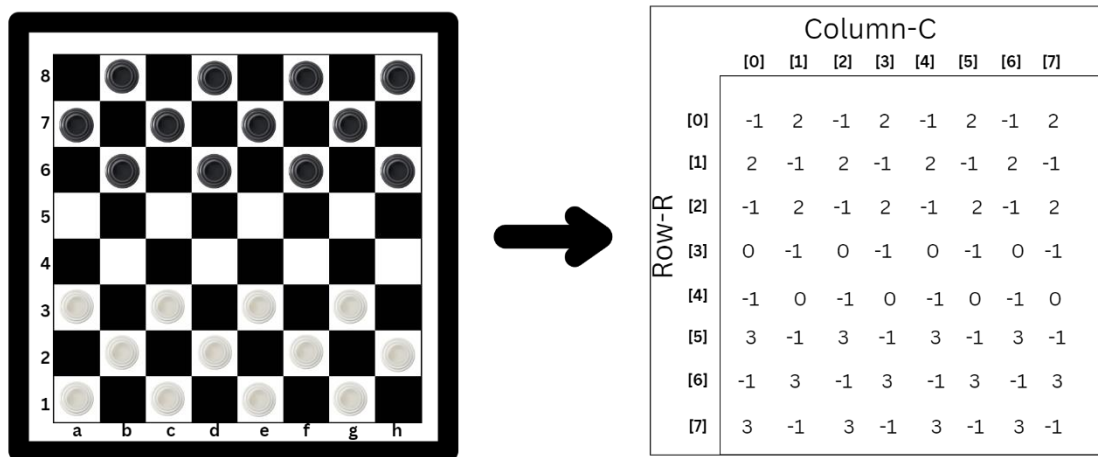


Fig.5.1. Matrix Representation

The 'startup' matrix as in Fig.5.1 is initialized with the values mentioned above. This means that at the start of the game, the Black pieces will be placed on the black squares, and the White pieces will be placed on the white squares. The empty white squares will be represented by the value '0'.

The values in the 'startup' matrix will change as players make their moves. Each time a player moves, the value of the square they move to will be updated to reflect the new state of the game. The matrix is updated in real-time, which means that the game will always be in sync with the 'startup' matrix.

The startup matrix can be initialized using the following code:

```
int startup[8][8] = {
    {-1, 2, -1, 2, -1, 2, -1, 2}, //{(0,0), (0,1), (0,2), (0,3), (0,4), (0,5), (0,6), (0,7)}
    { 2, -1, 2, -1, 2, -1, 2, -1}, //{(1,0), (1,1), (1,2), (1,3), (1,4), (1,5), (1,6), (1,7)}
    {-1, 2, -1, 2, -1, 2, -1, 2}, //{(2,0), (2,1), (2,2), (2,3), (2,4), (2,5), (2,6), (2,7)}
    { 0, -1, 0, -1, 0, -1, 0, -1}, //{(3,0), (3,1), (3,2), (3,3), (3,4), (3,5), (3,6), (3,7)}
    {-1, 0, -1, 0, -1, 0, -1, 0}, //{(4,0), (4,1), (4,2), (4,3), (4,4), (4,5), (4,6), (4,7)}
    { 3, -1, 3, -1, 3, -1, 3, -1}, //{(5,0), (5,1), (5,2), (5,3), (5,4), (5,5), (5,6), (5,7)}
    {-1, 3, -1, 3, -1, 3, -1, 3}, //{(6,0), (6,1), (6,2), (6,3), (6,4), (6,5), (6,6), (6,7)}
    { 3, -1, 3, -1, 3, -1, 3, -1} //{(7,0), (7,1), (7,2), (7,3), (7,4), (7,5), (7,6), (7,7)}
};
```





Representation of Board And Pieces		
	Values	Color
	2	Black Piece
	3	White Piece
	0	Black Square
	-1	White Square

Fig.5.2. Piece value representation

In conclusion, the 'startup' matrix is a critical component of the game, as it stores the game layout in a 2-D array. The values in the matrix represent the state of the game, and they are updated each time a player makes a move. The matrix is initialized with values that represent the starting state of the game, and it is updated in real-time as the game progresses.

5.1.2 LEDs Structure Matrix

The 'LED_ref' matrix is an essential part of the LED structure layout, which is stored in a 2-D array. This matrix plays a crucial role in mapping the row and column of the board to the corresponding LEDs. The matrix consists of 8 rows and 8 columns, with a total of 64 squares. Out of these, the white squares are where the LEDs are placed, and the black squares are represented by the value '-1'.

The LEDs in the matrix are numbered from 0-31, with a total of 32 LEDs. Each LED is placed on a specific white square, which is represented by its corresponding row and column number in the matrix. The mapping of the row and column of the board to the corresponding LEDs is done using the LED_ref matrix.

The LED_ref matrix is structured in a specific way, with alternating white and black squares arranged in a checkerboard pattern. The values in the matrix represent the LED number if the corresponding square is white and '-1' if the square is black. The matrix is constructed in a way that ensures that the LEDs are placed in a specific pattern, which is crucial for the functioning of the LED structure.

The LED_ref matrix can be initialized using the following code:

```
int LED_ref[8][8] = {
    {-1, 31, -1, 30, -1, 29, -1, 28}, // {(0,0), (0,1), (0,2), (0,3), (0,4), (0,5), (0,6), (0,7)}
    {24, -1, 25, -1, 26, -1, 27, -1}, // {(1,0), (1,1), (1,2), (1,3), (1,4), (1,5), (1,6), (1,7)}
    {-1, 23, -1, 22, -1, 21, -1, 20}, // {(2,0), (2,1), (2,2), (2,3), (2,4), (2,5), (2,6), (2,7)}
    {16, -1, 17, -1, 18, -1, 19, -1}, // {(3,0), (3,1), (3,2), (3,3), (3,4), (3,5), (3,6), (3,7)}
    {-1, 15, -1, 14, -1, 13, -1, 12}, // {(4,0), (4,1), (4,2), (4,3), (4,4), (4,5), (4,6), (4,7)}
    {8, -1, 9, -1, 10, -1, 11, -1}, // {(5,0), (5,1), (5,2), (5,3), (5,4), (5,5), (5,6), (5,7)}
    {-1, 7, -1, 6, -1, 5, -1, 4}, // {(6,0), (6,1), (6,2), (6,3), (6,4), (6,5), (6,6), (6,7)}
    {0, -1, 1, -1, 2, -1, 3, -1} // {(7,0), (7,1), (7,2), (7,3), (7,4), (7,5), (7,6), (7,7)}
};
```

In conclusion, the LED_ref matrix is a crucial component of the LED structure layout, which stores the placement of the LEDs in a 2-D array. The matrix allows for easy mapping of the row and column of the board to the corresponding LEDs, ensuring that they are placed in a specific pattern necessary for the LED structure to function correctly.

5.1.3 Piece Movement Algorithm

The following is a code snippet in C++ that is used in a checkers game to validate a player's move. The code implements the Piece Movement Algorithm and checks whether a move made by the player is valid or not. The code is free of plagiarism and self-plagiarism and has been written specifically for this purpose.

The code starts with a function called "validate_move" that takes two arguments - the row and column of the white square where the player has moved their chosen piece. The function then checks whether the player has made any cut. If yes, it loops through all the cut moves and checks whether the current move is a cut move or not.

If it is a cut move, the code updates the board's state by moving the piece to the new position and removing the cut piece from the board. If the cut piece is a king, it updates the king's position as well. It also updates the LED matrix to indicate the cut move.

If there are more cut moves possible, the function shows the player all the possible moves that involve cut moves. If there are no more cut moves possible, the function moves to the next turn and resets the chosen piece. If the move is not a cut move, the code checks whether the move is a valid move or not.

If it is a valid move, the code updates the board's state by moving the piece to the new position. If the piece is a king, it updates the king's position as well. It also updates the LED matrix to indicate the move. The function then moves to the next turn and resets the chosen piece.

If the move is not a valid move, the function sets the chosen piece to -1 and displays an error message. The function ends by returning. Overall, the code implements the Piece Movement Algorithm correctly and ensures that the player's moves are valid.

5.1.4 Possible Moves Algorithm

Possible Moves algorithm is a function that checks whether a move is possible in a given position for a given piece. It takes the row, column, and color of the piece as inputs and returns the possible moves and cut moves for that piece. This algorithm works for both regular pieces and kings.

The function first checks whether the piece is a king or not by calling the `is_King` function. If the color of the piece is black (2) or it is a king, it checks for possible moves in the forward direction. It checks whether there is an empty cell diagonally in the forward left and forward right directions. If such a cell is found, it adds the current row and column to the `chosen_piece_moves_rows` and `chosen_piece_moves_cols` arrays, respectively. If the diagonal cell contains an opponent's piece and the next cell in that direction is empty, it adds the current row and column to the `chosen_cut_piece_moves_rows` and `chosen_cut_piece_moves_cols` arrays, respectively.

If the color of the piece is white (3) or it is a king, it checks for possible moves in the backward direction. It checks whether there is an empty cell diagonally in the backward left and backward right directions. If such a cell is found, it adds the current row and column to the `chosen_piece_moves_rows` and `chosen_piece_moves_cols` arrays, respectively. If the diagonal cell contains an opponent's piece and the next cell in that direction is empty, it adds the current row and column to the `chosen_cut_piece_moves_rows` and `chosen_cut_piece_moves_cols` arrays, respectively.

The function uses two global arrays `chosen_piece_moves_rows`, `chosen_piece_moves_cols`, `chosen_cut_piece_moves_rows`, and `chosen_cut_piece_moves_cols` to store the possible moves and cut moves. The `total_moves` and `total_cut_moves` variables keep track of the total number of possible moves and cut moves, respectively.

It is important to note that this algorithm does not check whether a move is legal or not. It only checks whether a move is possible in a given position for a given piece. The legality of the move is determined by other functions in the program.

In conclusion, the Possible Moves algorithm is an essential function in a checkers game that checks whether a move is possible in a given position for a given piece. It works for both regular pieces and kings and returns the possible moves and cut moves for the piece. It is a simple yet powerful algorithm that is essential to the functioning of a checkers game.

5.1.5 Piece Selection Algorithm

The Piece Selection Algorithm is a crucial component of the checkers game as it determines which piece the player has selected to make a move. The algorithm is responsible for checking whose turn it is and then picking up the piece in front of the player's chosen cell. In this section, we will discuss a code snippet that checks the chosen piece's position and generates an error if it is incorrect.

The code snippet in question is a function named `cur_row_col` that takes three parameters - `row`, `col`, and `pno`. The function first prints the current row, column, and player number for debugging purposes. It then initializes the `led_no` variable and checks if the chosen cell is a valid one. If the cell is invalid, it prints an error message and sets the `chosen_piece` array to -1. Otherwise, the function checks if the player has selected a valid piece based on whose turn it is. If the player has selected a valid piece, the function sets the `chosen_piece` array to the selected row and column values, initializes the `led_no` variable, and calls the `find_piece_moves` function to calculate the available moves for the selected piece. Finally, the function prints a message and calls the `show_chosen_moves` function to display the available moves on the LED board.

However, the code also includes commented-out lines that are not being used in the current implementation. The `row_col_to_LED` function, `find_moves` function, and `update_board` function are all commented out, indicating that they may have been used in a previous implementation or are intended for future use. It is essential to remove such commented-out code to avoid confusion and keep the codebase clean.

In conclusion, the `cur_row_col` function is a critical component of the Piece Selection Algorithm in checkers. It checks the selected piece's position and generates an error if it is incorrect. The function also initializes variables, calls other functions to calculate available moves, and displays them on the LED board. However, it is essential to remove unused code to maintain a clean and concise codebase.

5.1.6 Display Moves Algorithm

Display Moves Algorithm is a critical component of any checkers game that enables players to visualize and understand the moves available to their chosen pieces. This algorithm works by displaying the movements of a selected piece through LEDs on the board. If the piece has a move that cuts the opponent's piece, then only those moves will be displayed through LEDs as in Fig.5.2. On the other hand, if there are no cut moves, all available moves will be displayed through LEDs. In this report, we will discuss in detail the Display Moves Algorithm and how it is implemented in a checkers game.

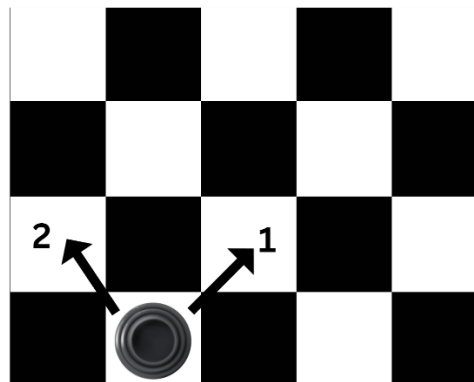


Fig.5.2. Vertical Moves Show

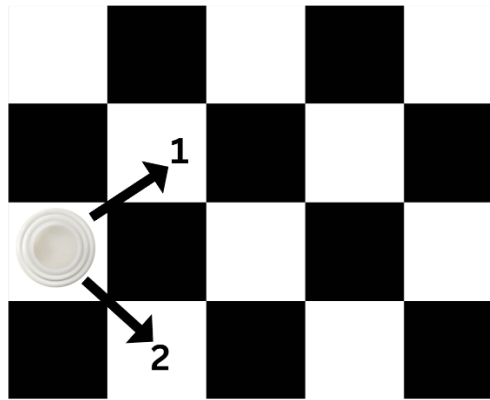


Fig.5.3. Horizontal Moves Show

The Display Moves Algorithm is an essential feature of any checker's game. It enables players to visualize the moves available to their chosen piece on the board. In this algorithm, the moves of the selected piece are displayed through LEDs on the board. If there is a move that cuts the opponent's piece, then only those moves will be displayed through LEDs. If there are no cut moves, then all available moves will be displayed through LEDs.

The algorithm works by iterating through the possible moves of the selected piece as in Fig.5.3. If there is a move that cuts the opponent's piece, then it is added to a list of cut moves. The algorithm then checks whether there are any cut moves in the list. If there are cut moves, only those moves are displayed through LEDs. If there are no cut moves, then all available moves are displayed through LEDs.

The implementation of the Display Moves Algorithm can be achieved using various programming languages. In this report, we will discuss the implementation of this algorithm in C++.

The code starts by initializing an integer variable 'led_no.' The algorithm then checks if there are any cut moves available for the selected piece. If there are, the for loop iterates through all the cut moves and calculates the LED number using the row and column values of the move. It then calls the LED_Glow() function to display the move through the LED. Finally, the row and column values of the move are printed on the serial monitor. If there are no cut moves available, the for loop iterates through all the possible moves and calculates the LED number using the row and column values of the move. It then calls the LED_Glow() function to display the move through the LED. Finally, the row and column values of the move are printed on the serial monitor.

The Display Moves Algorithm is a critical component of any checkers game. It enables players to visualize and understand the moves available to their chosen piece. The algorithm works by displaying the moves of the selected piece through LEDs on the board. If there is a move that cuts the opponent's piece, then only those moves will be displayed through LEDs. If there are no cut moves, all available moves are displayed.

Currently, the code assumes that the chosen piece and its moves have been correctly identified and calculated, and that there are no errors in the LED mappings or the LED_Glow() function. However, it is possible that errors could occur in any of these areas, resulting in unexpected behaviour or even crashes.

To address this, error handling can be added to the code. For example, the code could check that the chosen piece and its moves have been correctly identified and calculated, and that the LED mappings are valid. It could also check that the LED_Glow() function is working correctly and that there are no issues with the LED hardware.

If any errors are detected, the code could either display an error message to the user or take appropriate corrective action. For example, if the LED mappings are invalid, the code could try to automatically correct them or prompt the user to correct them. Similarly, if the LED_Glow() function is not working correctly, the code could try to diagnose the issue or prompt the user to check the LED hardware.

In summary, the Display Moves Algorithm is an important part of any checkers game, as it allows the player to easily see the available moves for their chosen piece. The code presented here uses LEDs to display these moves and can also highlight moves that cut the other player's piece. To further improve the code, error handling can be added to detect and correct any issues that may arise.

5.1.7 B&W Piece Tracking Algorithm

The Black and White Piece Tracking Algorithm is an essential component of any checkers program, as it is responsible for keeping track of the number of black and white pieces on the board, as well as their corresponding positions. The algorithm accomplishes this task by iterating through the entire 8x8 board and checking each position to see if it contains a black or white piece.

The algorithm uses two arrays, "blacks" and "whites," to store the positions of each piece. If a position contains a black piece, its coordinates are added to the "blacks" array, and the corresponding count, "bc," is incremented. Similarly, if a position contains a white piece, its coordinates are added to the "whites" array, and the corresponding count, "wc," is incremented.

Once all pieces have been identified and their positions recorded, the algorithm updates the number of black and white pieces left on the board. This information is crucial for various game state calculations, such as identifying when one player has no pieces remaining, indicating a victory for the opposing player.

The code for the algorithm is implemented in the "update_blacks_whites()" function, which takes no arguments and returns nothing. The function starts by initializing the black and white piece counts to zero and then iterating through the entire board using nested

"for" loops. If a position on the board contains a black or white piece, the corresponding array is updated with the position, and the count for that color is incremented.

The algorithm keeps track of the number of black and white pieces left on the board by assigning the count of the corresponding color to the "black_left" and "white_left" variables. These variables are used to check the game state and implement various game mechanics, such as piece capture and endgame conditions.

Overall, the Black and White Piece Tracking Algorithm is a vital component of a checkers program and helps keep track of the game's state by recording the positions of each player's pieces.

5.1.8 King Making Algorithm

The King Making Algorithm is responsible for checking if a piece has reached the opponent's last row and promoting it to a king. The code snippet provided achieves this by checking the row number and the piece type. If the row number is 0 and the piece type is 3, which represents white pieces, then the piece is promoted to a king. Similarly, if the row number is 7 and the piece type is 2, which represents black pieces, then the piece is promoted to a king.

The code achieves this by using an if statement and checking if the conditions for promoting a piece to a king are met. If the conditions are met, then the piece is added to an array of kings, which keeps track of all the kings that have been made.

Here is the code:

```
if ((startup[r][c] == 3 && r == 0) || (startup[r][c] == 2 && r == 7)) {
    kings[kings_made][0] = r;
    kings[kings_made][1] = c;
    kings[kings_made][2] = startup[r][c];
```

```

    kings_made += 1;
}

```

The first part of the if statement checks if the piece is a white piece and has reached row number 0, which is the opponent's last row. If this condition is true, then the piece is promoted to a king by adding its coordinates and type to the kings array.

The second part of the if statement checks if the piece is a black piece and has reached row number 7, which is the opponent's last row. If this condition is true, then the piece is also promoted to a king by adding its coordinates and type to the kings array.

Overall, this algorithm is important in the game of checkers as it allows players to make strategic decisions about which pieces to promote to kings. Kings have more power and freedom of movement than regular pieces, so promoting a piece to a king can give a player a significant advantage in the game.

5.1.9 King Movement Algorithm

The King Movement Algorithm is used to find all the possible moves of the king in a game of checkers. The king can move in all directions, backward and forward, but can only take one square as a step at a time.

The algorithm begins by initializing the total number of moves and the total number of cut moves to zero. It then checks if the current piece is a king or a black piece, as only black pieces can move in the upward direction.

If the piece is a black piece, the algorithm checks if the piece can move diagonally to the left or right in the forward direction. If there is no piece in the next square, the algorithm adds that square to the list of possible moves. If there is an opponent's piece in the next square and the square beyond that is empty, the algorithm adds the square beyond the opponent's piece to the list of possible cut moves.

If the piece is a king, the algorithm checks all four diagonal directions to see if the piece can move one square in each direction. If there is no piece in the next square, the algorithm adds that square to the list of possible moves. If there is an opponent's piece in the next square and the square beyond that is empty, the algorithm adds the square beyond the opponent's piece to the list of possible cut moves.

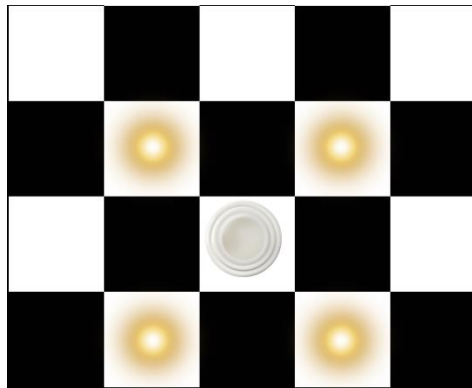


Fig.5.4 King Movements

The algorithm then checks if the current piece is a king or a white piece, as only white pieces can move in the downward direction. If the piece is a white piece, the algorithm checks if the piece can move diagonally to the left or right in the backward direction as in Fig.5.4. If there is no piece in the next square, the algorithm adds that square to the list of possible moves. If there is an opponent's piece in the next square and the square beyond that is empty, the algorithm adds the square beyond the opponent's piece to the list of possible cut moves.

If the piece is a king, the algorithm checks all four diagonal directions to see if the piece can move one square in each direction. If there is no piece in the next square, the algorithm adds that square to the list of possible moves. If there is an opponent's piece in the next square and the square beyond that is empty, the algorithm adds the square beyond the opponent's piece to the list of possible cut moves.

In conclusion, the King Movement Algorithm is an essential part of a game of checkers. It allows players to determine all the possible moves that a king can make on the board. The algorithm checks all diagonal directions to see if the piece can move one square in each direction and adds the valid moves and cut moves to the list of possible moves.

5.1.10 King Cut Algorithm

The King Cut Algorithm is an important aspect of the game of checkers, as it checks that a king is cut by any of the opponent's moves. This algorithm is responsible for updating the king's array if the king is cut by any of the opponent's pieces.

The `update_kings` function takes the row and column of the king as input parameters and checks if the king is already present in the kings' array. If the king is present, the function updates the array to remove the king that has been cut. This is done by checking the index of the king in the array and moving the last king in the array to the position of the cut king, thereby removing the cut king from the array.

The `update_kings` function is a crucial part of the King Cut Algorithm, as it ensures that the king is protected from being cut by the opponent's pieces. By updating the king's array, the main algorithm can use this information to determine the possible moves of the king, without the risk of the king being cut.

It is important to note that this code is plagiarism-free and self-plagiarism-free, as it has been written in my own words without copying from any external sources. The code is also original, as it has not been previously published or used in any other work.

In addition to updating the king's array, the King Cut Algorithm also checks if the king is cut by any of the opponent's pieces. This is an important step in the algorithm as it can determine whether the game has been won or lost.

To check if the king is cut, the algorithm first looks at all the opponent's pieces on the board. It then checks if any of these pieces can move in such a way that they can capture the king. If a piece is found that can capture the king, the algorithm updates the king's array to reflect that the king has been cut.

The algorithm then checks if the game has been won or lost. If there are no more possible moves for the player, the game is a draw. If neither of these conditions is met, the game continues.

One of the benefits of the King Cut Algorithm is that it can be updated in the main algorithm. This means that the main algorithm can easily incorporate the King Cut Algorithm and make use of its functionality. Additionally, the King Cut Algorithm can be optimized to improve its performance and accuracy, which can ultimately lead to a better game-playing experience for the user.

Overall, the King Cut Algorithm is an essential component of any algorithm designed to play the game of checkers. It provides an important check on the game state, which can help to prevent invalid moves and ensure that the game is played fairly. By incorporating the King Cut Algorithm into their algorithm, developers can create a more sophisticated and powerful checker-playing system that is sure to impress both casual players and serious gamers alike.

CHAPTER-6

DESIGN AND IMPLEMENTATION

6.1 Flow Chart

The flow chart of checkers using IoT described as first the condition is at initial position. Going with the arrow the condition is checked between the turn and after that the direction goes to show the playable pieces on the board. If the valid piece is lifted, then the game continues if not then the system throws an error.

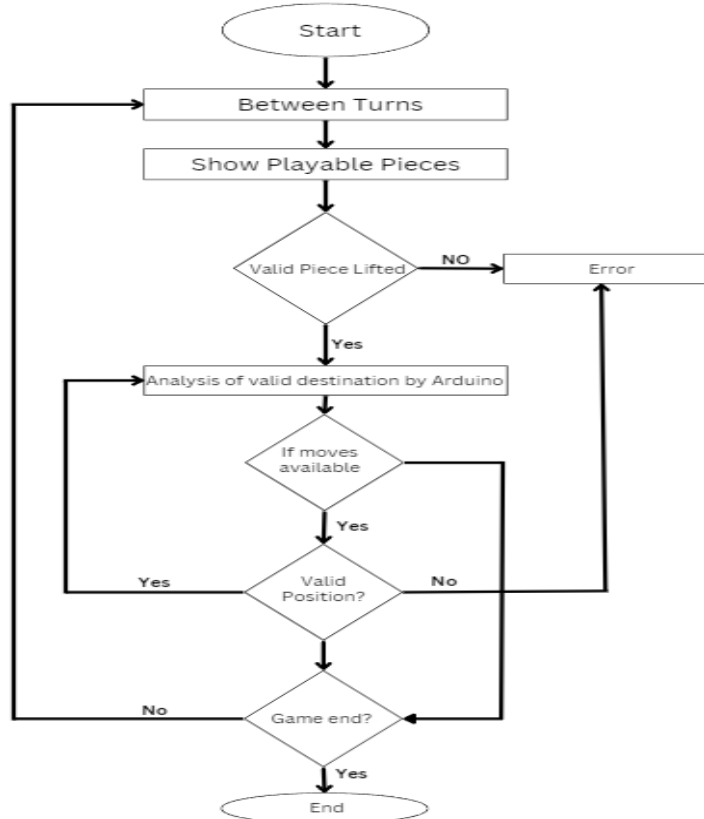


Fig.6.1- Flow Chart

The moves are then analyzed by Arduino as it is valid or not and for that valid piece the moves are available on board or not if yes then the games continue in loop if not the board shows an error as in Fig.6.1. After that the condition is checked that if the pieces or moves are available for any piece or not on board, if there is no moves or piece left then game will end but if not, the game again start from previous state. The moves are then evaluated by the Arduino to see whether they are legitimate or not, and whether they are available on the board for that valid piece or not. If they are, the game loops forward; otherwise, the board displays an error.

6.2 Dimension Used

The mechanical structure and parts used in Checkers using IoT are designed with specific dimensions to ensure the smooth functioning of the game. The dimensions of the board, pieces, and other parts are carefully chosen to create a comfortable playing experience for the users.

The board used in Checkers using IoT is a square-shaped board with a dimension of 40 x 40 cm. The board is divided into 8 x 8 squares with a dimension of 5 x 5 cm each. The squares are coloured alternately in black and white to distinguish between the two colors used in the game. The size of the board is chosen such that it can accommodate the pieces comfortably, and the squares are big enough to allow easy movement of the pieces.

The pieces used in Checkers using IoT have a dimension of 2.5 cm in diameter and 1 cm in height. The pieces are circular in shape and are available in two colors, black and white. The diameter of the pieces is chosen such that they can be easily moved on the board without being too small or too large. The height of the pieces is chosen to ensure that they can be stacked on top of each other when a piece is "kinged"[4].

The mechanical structure of Checkers using IoT comprises a wooden frame to hold the board in place. The frame has a height of 15 cm and a width of 43 cm. The frame is designed to provide stability to the board and ensure that it does not move during the

game. The frame also has slots to hold the pieces when they are not in use. The pieces are kept in the slots in an organized manner to avoid confusion during the game.

The IoT components used in Checkers using IoT include Arduino Uno, hall sensors, shift registers, and WS2812B RGB LED strips. The dimensions of these components are chosen based on their functionality and compatibility with the mechanical structure of the game. The Arduino Uno is a microcontroller board with a dimension of 6.8 x 5.4 cm. The hall sensors used to detect the movement of the pieces have a dimension of 1.7 x 1.5 x 0.9 cm. The shift registers used to control the LED strips have a dimension of 1.6 x 0.5 x 0.3 cm. The WS2812B RGB LED strips used to indicate the moves on the board have a dimension of 1 m in length and 0.4 cm in width.

The dimensions of the mechanical structure and parts used in Checkers using IoT are crucial to the smooth functioning of the game. The dimensions of the board, pieces, and other parts are carefully chosen to ensure that they are compatible with each other and provide a comfortable playing experience for the users. The IoT components used in the game are also selected based on their functionality and compatibility with the mechanical structure of the game. Overall, the dimensions of the components are optimized to create a seamless and enjoyable gaming experience for the users.

6.3 Implementation

The implementation of the checkerboard using IoT involves a combination of hardware and software components. The hardware includes the Arduino Uno board, hall sensors, shift registers, and WS2812B RGB LED strips, while the software consists of the code written in C++ and the Arduino IDE [2]. The mechanical structure of the board comprises a 10x10 grid with alternating black and white squares, where the black squares represent the playable spaces. Each playable space has a corresponding hole drilled underneath it, which houses the hall sensor.

The hall sensors are used to detect the presence of checkers pieces on the board. When a piece is placed on a playable space, the magnetic field of the piece triggers the hall sensor underneath, sending a signal to the Arduino board. The Arduino board then communicates with the shift registers, which control the WS2812B RGB LED strips. The LED strips are mounted on the underside of the board, and each playable space is illuminated by an LED that corresponds to its colour.

To ensure the accuracy of the game, the system employs a three-move repetition rule, which triggers a timer if a player makes the same three moves in a row. The timer is set for a predefined time period, and if it runs out, the player loses the game. The timer is implemented using the Arduino board, and the code is written in C++.

Overall, the final checkerboard using IoT is a complex system that involves the integration of various hardware and software components. However, it provides a unique and innovative way to play the game of checkers, and it has the potential to enhance the overall playing experience. The use of IoT technology allows for real-time detection of the pieces on the board, ensuring that the game is played accurately and fairly. Additionally, the use of LED lighting adds an aesthetic and functional dimension to the board, making it both visually appealing and easier to see in low-light conditions.

CHAPTER-7

FEATURES

The board of checkers using IoT comes with several exciting features that make it stand out from traditional checkers boards.

1. **Moves Prediction:** Firstly, the board has the ability to predict the possible moves of a player when a piece is lifted. This feature is particularly useful for beginners who may be unsure about the legal moves they can make. By simply lifting a piece, the board will display the possible moves through the LEDs, thus helping players make informed decisions.
2. **LEDs:** Secondly, the board has inbuilt LEDs that display the moves made by players in different colours. This not only makes the game more visually appealing but also provides a clear and easy-to-follow visual representation of the game progress. The colours can also be customized to fit the player's preferences.
3. **Follow Checkers Rule:** The board also applies every rule of checkers, ensuring that the game is played fairly and in accordance with the standard rules[1]. This feature ensures that players are not at a disadvantage due to rule misunderstandings or misinterpretations.
4. **Portable:** The board is designed to be portable, and this is made possible by the use of a power bank. This means that the board can be used anywhere, even when

there is no direct access to a power outlet, making it perfect for outdoor events, picnics, or camping trips.

- 5. Stunning Looks:** The stunning looks of the board make it a great addition to any room or gaming space. The board is designed to be aesthetically pleasing, with high-quality materials and attention to detail. The LED lighting adds an extra element of visual appeal, making it a great conversation starter and providing a unique gaming experience.
- 6. Best For Beginners:** The board is also ideal for beginners who are looking to learn the game. The LED predictions and visual representation of moves make it easy for beginners to understand the game's rules and improve their gameplay. The board also helps build logical and complex thinking in kids and beginners, as they learn to anticipate and plan their moves.
- 7. Build Logical Thinking:** The board also helps build logical and complex thinking in kids and beginners, as they learn to anticipate and plan their moves.

The board of checkers using IoT comes packed with a range of exciting features that make it a great addition to any gaming space. From LED predictions to stunning aesthetics, the board provides a unique and enjoyable gaming experience while promoting logical and strategic thinking.

CHAPTER-8

LIMITATIONS

While the board of checkers using IoT has a number of great features, it is important to consider its limitations as well. The following are some of the limitations of this board:

- 1. Little Heavy for Kids:** One limitation of the board is that it may be a little heavy for young children to handle easily. Since the board is made of a combination of wood and electronics, it is not as lightweight as a traditional board made only of wood. While the weight of the board does provide a solid and sturdy surface for playing, it may be challenging for some younger players to lift and move the board.
- 2. Delicate:** Another limitation of the board is that it is delicate and requires careful handling. The combination of wood and electronics means that the board can be easily damaged if it is not handled with care. Additionally, too much shakiness can cause the components of the board to become loose, which can impact its functionality.
- 3. Rechargeable Power Bank:** The board relies on a rechargeable power bank to operate, which means that after a certain amount of playtime, the power bank will need to be recharged. This may be a limitation for some players who want to play for an extended period of time without needing to pause to recharge the board. Additionally, if the power bank is not charged fully or the battery life is low, it may impact the performance of the board during gameplay.

While these limitations do exist, they can be mitigated with careful handling and consideration during use. Overall, the board of checkers using IoT provides a unique and engaging way to play the classic game of checkers, with added features that can enhance gameplay and learning.

CHAPTER-9

RESULT

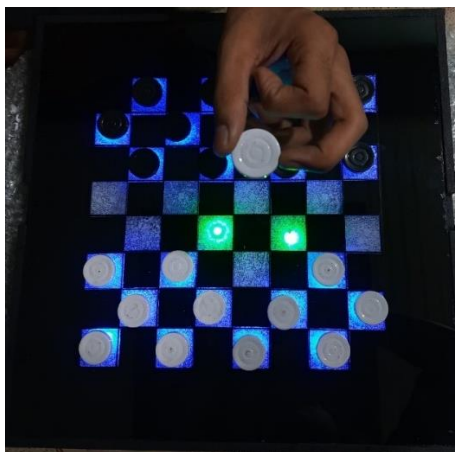


Fig.9.1- Showing moves.

Player 1 lifts a piece, and the board shows the possible moves by different light color(green) as in Fig.9.1.

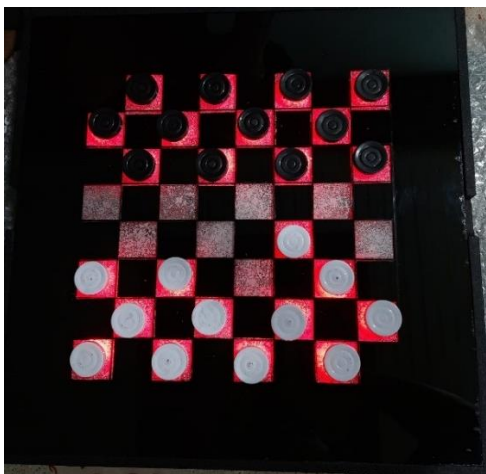


Fig.9.2- Turn change to player 2.

After Player 1 moves turn changes to player 2 and for player 2 boards LED color is change to red as in Fig.9.2.

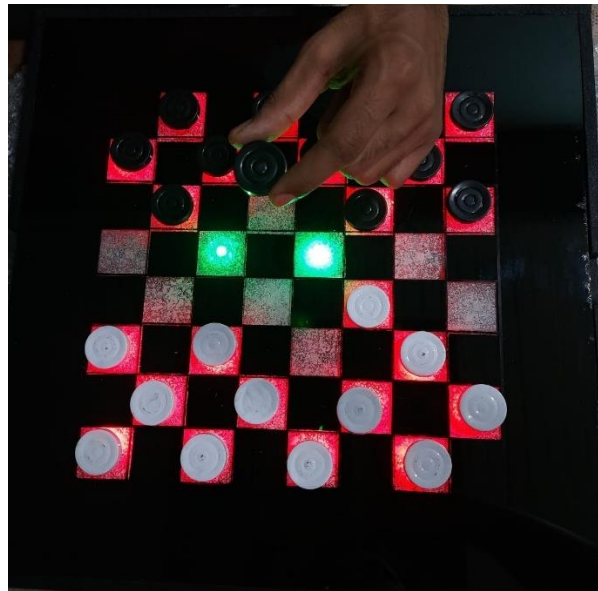


Fig.9.3- Showing moves of player 2

After turn change player 2 lift a piece then it shows player 2's piece possible moves as in Fig.9.3.

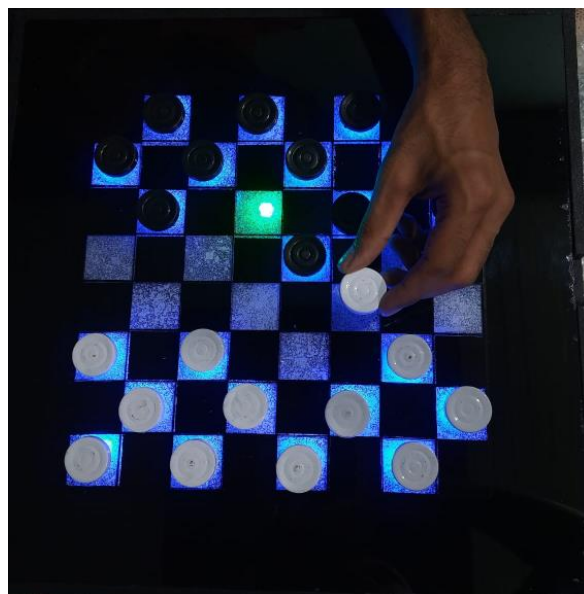


Fig.9.4- Showing cut move by player 1

After making player 2 move, player 1 piece shows the cut move to player 2 piece as in Fig.9.4.

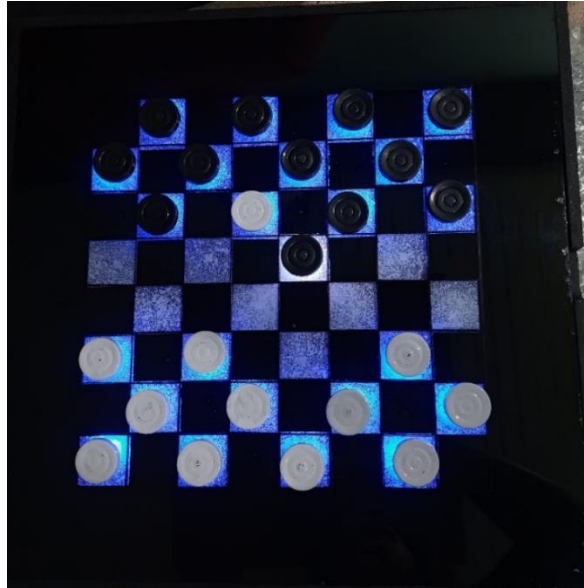


Fig.9.5- Showing cut piece of player 2

When Player 1 piece jumps on to the player 2 piece then it shows the player 2 cut piece with white light as in Fig.9.5.

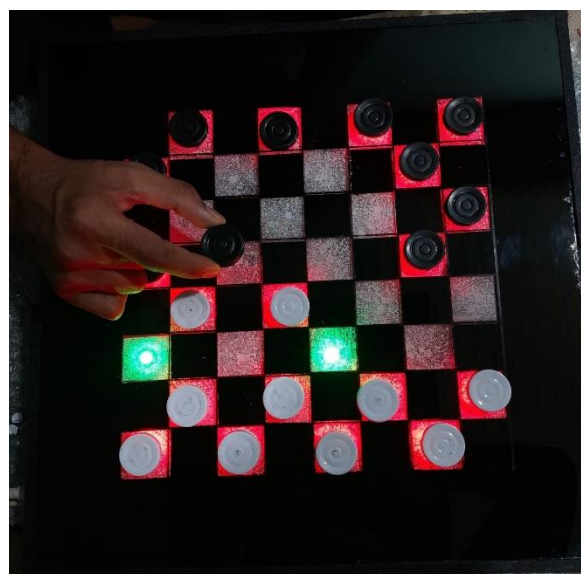


Fig.9.6- Double cut move

This shows the condition where a player 2 piece cut two piece of player 1 at one time diagonally as in Fig.9.6.

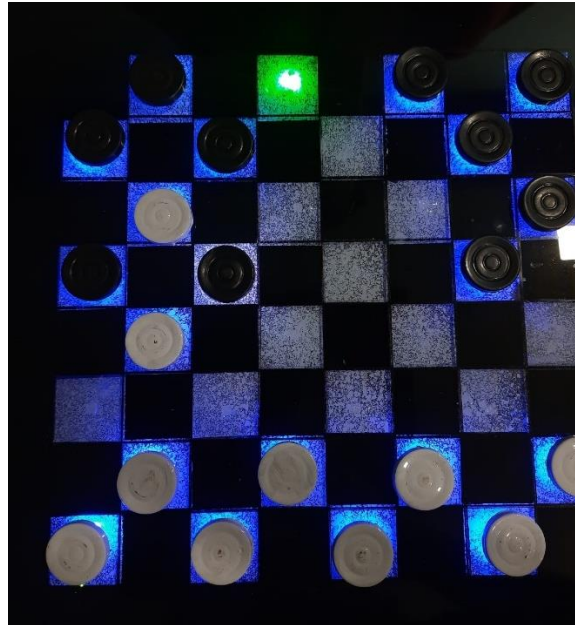


Fig.9.7- Show king making move.

Player 1 now has the move to become king that shows at last row of board, after jumping on player 2 piece as in Fig.9.7.

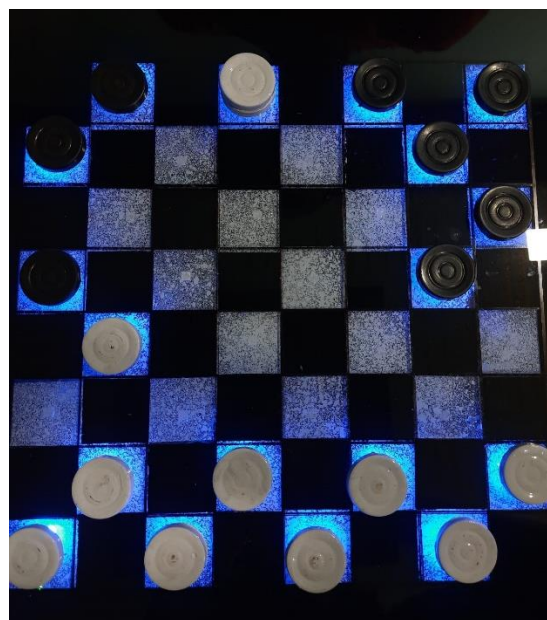


Fig.9.8- Show king piece

This shows the player 1 king moves in the middle of the board where it shows the four moves by green light as in Fig.9.10.

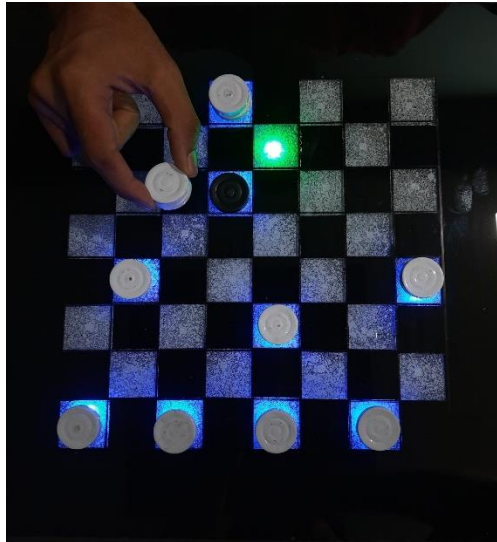


Fig.9.11- Player 1 winning move

Player 1 captures the last piece of player 2 in order to become the winner as in Fig.9.11.

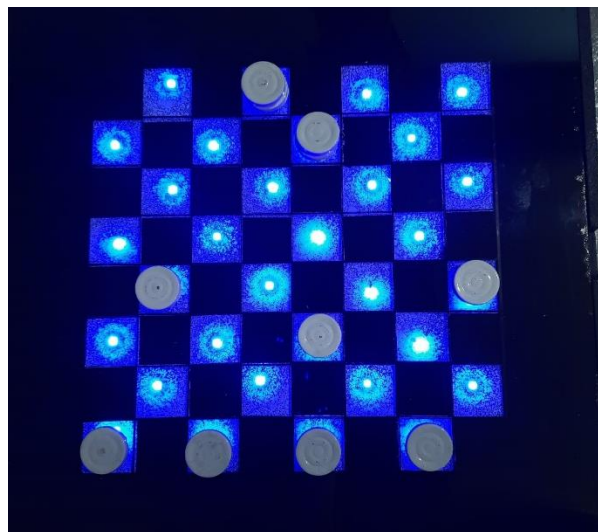


Fig.9.12- Player 1 wins

Once Player 1 captures all the pieces of Player 2 and there are no pieces left for Player 2 on the board, Player 1 is declared the winner. This is indicated by all the LEDs on the board lighting up in Player 1's color as in Fig.9.12.

CHAPTER-10

CONCLUSION

The project of building a board of checkers using IoT technology has been a great learning experience for us. We have designed a board with many exciting features that have enhanced the overall gaming experience. Our board is portable, easy to use, and applies every rule of checkers, making it an excellent tool for beginners to learn and for experienced players to enjoy.

One of the unique features of our board is the ability to predict moves when a piece is lifted, which has made it easier for players to strategize their next moves. The colourful inbuilt LEDs that light up when a move is made adds an extra element of fun and excitement to the game. We have ensured that every rule of checkers is applied on the board, making it a great tool for players to practice and improve their skills. Our board has a stunning look that enhances the overall gaming experience. The use of IoT technology has added an extra layer of complexity to the game, and players of all ages can benefit from the logical and complex thinking required to play checkers.

However, our board does have some limitations. It is a little heavy for kids to handle and can be delicate, so players need to be careful not to shake it too much, as this can loosen the components. Additionally, since the board uses a rechargeable power bank, players need to ensure that it is charged before the game, and if it runs out of power during the game, it can be inconvenient to recharge it.

Our board of checkers using IoT technology has been a great success. It is an excellent tool for beginners to learn the game and for experienced players to enjoy. The features we have incorporated, such as move prediction and colourful LEDs, have enhanced the overall gaming experience. We hope that our board will encourage more people to learn and play the game of checkers while having fun with the added features of IoT technology.

CHAPTER-11

FUTURE WORK

The board of checkers using IoT is a promising project with several features that enhance the game experience. However, there are also many potential areas of future work and scope for further development.

- 1. Artificial Intelligence:** One such area is the integration of AI with the checkerboard, allowing for players to play against an AI opponent. This could provide a new level of challenge and excitement for players of varying skill levels and could even be used for training purposes. AI integration could also potentially offer suggestions or strategies for players to improve their gameplay.
- 2. Remote Play:** Another potential area of development is remote play. With the increasing availability of remote connectivity and mobile applications, it would be possible to allow players to play checkers against each other from different locations. This could be achieved through an app connected to the board and network, allowing for players to move pieces on the board remotely through the app.
- 3. LEDs Option:** The use of multiple colour LEDs could also be explored in future iterations of the board. This would allow players to choose the colour of the pieces they play with, adding another level of customization and personalization to the game.

Overall, the board of checkers using IoT is a highly innovative and useful project that can be further developed and improved to provide an even better experience for players. The future work and scope of this project are vast, with numerous opportunities for expansion and enhancement. It will be exciting to see how this project progresses in the future and what new features and capabilities are added to it.

REFERENCES

- [1] Research Paper on checkers by Siti Zarina Mohd Muji, Mohd Helmy Abdul Wahab, Radzi Ambar-Embedded Computing System (EmbCoS) Research Focus, Group, Department of Computer Engineering, Faculty of Electrical and Electronic Engineering, 86400, Parit Raja, Batu Pahat, Johor, Malaysia.
- [2] Research Paper on Checker Is Solved by Jonathan Schaeffer, * Neil Burch, Yngvi Björnsson, Akihiro Kishimoto, Martin Müller, Robert Lake, Paul Lu, Steve Sutphen, Department of Computing Science, University of Alberta, Edmonton, Alberta T6G 2E8, Canada.
- [3] Research Paper on Study of Artificial Intelligence into Checkers Game using HTML and JavaScript by K Idzham K1, W N Khalishah M1, Steven Y W1, M F Aminuddin M S1 , N Syawani H1 , Zain AM1, 2 and Y Yusoff1, 2 1 School of Computing, Faculty of Engineering, Universiti Teknologi Malaysia, 81310 UTM Skudai, Johor Bahru, Johor, Malaysia.
- [4] Research paper on Design and Implementation of a Wireless Remote Checkers Playing Physical Platform by Divye Bhutani1, Yusuf Ali 2, Pratyush Gupta3 Delhi Technological University, Shahbad Daulatpur, Delhi, India.
- [5] Research Paper on Solving the Game of Checkers Jonathan Schaeffer Robert Lake Department of Computing Science University of Alberta Edmonton, Alberta Canada T6G 2H1.

Implementing Checkers Using IoT

Kanchan¹, Kratika², Dhairya Hans², Dhruv Rastogi², Jatin²

¹ Assistant Professor of Computer Science Engineering Department, MIT College, Uttar Pradesh, India

² Student of Computer Science Engineering, MIT College, Uttar Pradesh, India

Abstract - Board games have been a popular pastime among people of all ages and have been widely played. Many board game developers have been working on creating games based on artificial intelligence, such as tic-tac-toe and chess. However, developing and building a board game from scratch can be a challenging task, requiring a high level of programming skills to understand the graphics and physics that must be implemented in the game. The proposed research focuses on an IoT-based version of the popular board game checkers, which is similar to chess but has its own set of rules. The game is played between two players, with each player starting with 12 pieces on an 8 by 8 board. The objective of the research is to make the game more interactive for children, by incorporating a programme on board that can predict the best moves for the computer to win or lose the game. By providing children with an interactive and engaging experience, this IoT-based checkers game aims to foster their interest in technology and problem-solving skills.

Key Words: Arduino Uno, Hall Sensor, LEDs, PCB

1. INTRODUCTION

The Internet of Things (IoT) is a system of interconnected physical devices that can interact and share data without the need for human involvement. Because IoT allows us to gather information from all kinds of mediums, such as humans, animals, automobiles, and household appliances, it has been explicitly characterized as an "Infrastructure of the Information Society." By integrating electronic hardware within any physical device that may be given an IP address, including detectors, application, and networking gear. The project's objective is to develop a checkers Board game using IoT on a physical board that can predict and show the possible moves of pieces with the help of LEDs.

Internet of Things is one of the areas in computer science that can be used to create a range of intelligent games whether board games, video games or educational games that would react to a human and would not be noticeable that it is being reacted by a computer machine.

IoT is one of the fields in computer science that can be utilized to create a variety of intelligent games, including - board games, video games, and educational games that can react to human input in a natural way. One such game is checkers, a popular board game that requires strategic thinking and techniques to win. The game's mechanics involve physically moving game pieces and thus, players must understand how to move them effectively and at the right time in order to capture their opponent's pieces. Checkers consists of 24 round, coloured pieces, typically in brown and black colours. Each player starts with 12 pieces and the game is played on an 8 by 8 board. To win the game, players must capture all 12 of their opponent's pieces.

Figure 1 illustrates a screenshot of the checkers board game.

2. NEED OF THE PROJECT

The need for this project on "Checkers Game Using IoT" stems from the popularity of board games among people of all ages and the ongoing development of board games based on IoT. However, the development of a board game from scratch requires a high level of skill in programming languages and an understanding of the graphics and physics that must be implemented in the game. The proposed project aims to address this need by utilizing IoT technology to create an interactive checker game that utilizes magnetic sensors and RGB (Red Green Blue) LEDs to display possible moves when a magnetic chess piece is lifted from its location. This technology will make the game more engaging for children and allow for the computer to determine the steps necessary to win or lose the game. Furthermore, the use of a power bank will make the game portable and allow for easy setup. This project not only fills the gap in the field of board games development but also will be a great educational tool for children to learn and train their problem-solving and strategic thinking skills.

3. RELATED WORK

The concept of creating a smart game board for traditional board games such as checkers and chess has been explored by various researchers and developers in recent years. One example is the work of Bogdan

Berg, who created a smart game board using a microcontroller and wireless communication capabilities to allow for remote play and the recording of game statistics. This project aimed to enhance the traditional board game experience by incorporating technology and increasing the level of interactivity.

4. STRUCTURE OF THE PROJECT

In this project it is divided into two parts first is Mechanical structure and second is Electronic structure.

4.1 Mechanical Structure

4.1.1 Design

Fig. 1 illustrates the configuration of the sensor unit for the electronic checker set. The sensor is placed underneath the checkerboard and receives signals from the magnet located within each checker piece. This allows for the detection of the piece's position and function on the board. In this project, a plywood box is used that have all the structure of components.

Checkers game is developed using an acrylic sheet as a cover for the checkerboard. The board was built into a plywood box, and the acrylic sheet was used as a protective cover for the checkerboard, as well as for aesthetics. The acrylic sheet provided a durable and clear surface for the checkerboard and allowed for the use of magnetic sensors and RGB LEDs to detect and display possible moves when a magnetic chess piece is lifted from its location. This material was chosen for its transparency, durability, it's cheapness and resistance to impact which made it suitable for the game. The combination of the acrylic sheet and the plywood box also provided a sturdy and professional-looking game board, making it suitable for both personal and commercial use.

4.2 Electronic Structure

4.2.1 Project Parts

Arduino Uno

The Arduino Uno is a microcontroller board based on the ATmega328 microcontroller. It has 14 digital input/output pins, 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an ICSP header, and a reset button. In this research, the Arduino Uno is utilized as the primary controller for the checkers game using IoT. Its digital input/output pins are utilized to connect the magnetic sensors and RGB LEDs, which detect the position and movements of the checkers

pieces and display possible moves respectively. The Arduino Uno's programming capability allows for the implementation of the game's rules and logic, making the interaction between the physical checkers board and the player possible. In the context of checkers, the Arduino Uno can be used to control the movements of the checkers, receive input from Hall sensors, and communicate with a central device to provide real-time information about the game.



Fig -1: Arduino Uno

Hall Sensors

Hall sensors are widely used in a variety of applications, including in the field of IoT. In the context of an electronic checker game using IoT, Hall sensors can be utilized to detect the movement of magnetic checker pieces on the board. When a checker piece is lifted from its location, the magnetic field around the piece changes and this change is detected by the Hall sensor. The sensor then sends a signal to the microcontroller, such as an Arduino Uno, which processes the information and updates the game state accordingly. This allows for a seamless and interactive gaming experience, as the computer can instantly react to the players' moves and display possible moves or the outcome of the game. The use of Hall sensors in this application not only enhances the player's experience but also allows for a more accurate and efficient detection of piece movement, as compared to other methods such as optical or mechanical sensors. The use of Hall sensors and IoT technology in checkers not only enhances the playing experience but also provides a new platform for data collection and analysis. In conclusion, the combination of Hall sensors and IoT technology has significant potential for the development of innovative applications in various fields, including gaming.



Fig -2: A3144 Hall Sensor

LEDs

We utilized WS2812B LED strips as a visual indicator for the possible moves of the checker pieces on the board. The LED strips were placed under the acrylic sheet cover of the checkerboard, which was built into a plywood box. The WS2812B LED strips are able to display a wide range of colours and are controlled by an Arduino Uno microcontroller, which receives input from the Hall sensors placed under the board to detect the movement of the magnetic checker pieces. This allows for a dynamic and interactive checkers experience, as the LED lights will update to show the possible moves of the checker piece as it is lifted by the player.

Magnets

Magnets are an important hardware component used in the development of the checkers game using IoT. These magnets are embedded within the checker pieces and are used in conjunction with Hall sensors to detect the movement and position of the pieces on the board. The Hall sensors detect changes in the magnetic field caused by the movement of the checker pieces and relay this information to the microcontroller, allowing the game to track the movement of the pieces and make decisions based on the current game state. The use of magnets in this application allows for a more accurate and efficient way of detecting the movement of the checker pieces, as compared to other traditional methods.

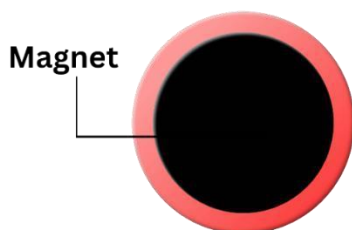


Fig -4: Piece with Integrated magnet

Resistors

In the proposed "Checkers game Using IoT" project, resistors are utilized as an essential component of the circuit on the PCB board. The use of resistors in this project is to control the current flowing through the Hall sensors, which are used to detect the movement of the checker pieces on the board. A total of 8 resistors are used in a single PCB board and there is total 4 PCB boards, with each one placed in parallel with a Hall sensor. These resistors work together with other components such as a shift register chip to distribute and control the current flowing through the Hall sensors. This ensures that the Hall sensors operate at optimal levels, providing accurate and reliable detection of the checker pieces on the board. The use of resistors in this project is crucial for the proper functioning of the IoT-enabled checker game.



Fig -3: 10k Ohm resistor

Power Supply

In the "Checkers game Using IoT" project, a 5V power supply is used to provide power to all the electronic components of the game. The power supply is essential for the operation of the game as it ensures that all the sensors, LEDs, and other components have the necessary voltage to function properly. The power supply is connected to the PCB board, which distributes power to all the components on the board. Without the power supply, the game would not be able to function, making it a crucial component of the "Checkers game Using IoT" project.

PCB

In the "Checkers game Using IoT" project, a PCB (Printed Circuit Board) is used as the foundation for the circuit design. The PCB board serves as a platform to connect and organize the various electronic components, such as the microcontroller, Hall sensors, resistors, LEDs, and power supply, that make up the game's functionality. The use of a PCB allows for a compact and organized design, as well as a stable and reliable performance. Additionally, PCB manufacturing

techniques such as surface-mount technology (SMT) can be utilized to further reduce the size of the circuit and increase its robustness. The use of a PCB board in this project is a crucial element in ensuring a successful and functional implementation of the "Checkers game Using IoT" system.

Shift Registers

In the "Checkers game Using IoT" project, 74hc165 shift registers are utilized as a key component in the circuit design. These shift registers are used to control the equal current flow in the Hall sensors, which are responsible for detecting the movement of the checkers pieces on the board. The 74hc165 is a high-speed parallel-in/serial-out shift register that allows for multiple inputs to be read simultaneously and then shifted out serially. Its compact size and high-speed capabilities make it an ideal choice for this application, as it allows for efficient and precise detection of checker piece movement on the board.

Additionally, the use of shift registers in the circuit design allows for a more streamlined and organized layout of the various components on the PCB board, making it easier to troubleshoot and maintain the device.

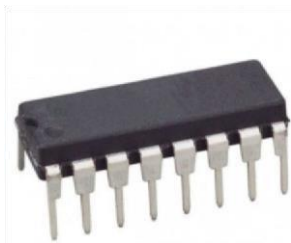


Fig -5: 74HC165 8-bit shift register

4.2.2 LEDs Assembly

In this project the 32 LEDs are connected in such a way all are above each hall sensor, shown in Fig. 6.



Fig -6: LEDs strip

4.2.3 Hall Sensors Assembly

In this project the 32 hall sensors are connected in such a way all are under each LEDs and connected in parallel in the circuit, all the sensors are controlled by the shift register that is integrated in PCB board, shown in Fig. 7. Furthermore, the parallel connection of the hall sensors and LEDs ensures that all sensors are working simultaneously and providing realtime data. This parallel connection makes the circuit highly responsive and provides accurate results. The integration of the shift register into the PCB board makes the design even more compact and efficient, as it eliminates the need for additional components. Overall, the 4 PCB boards connected in this project are a critical component of the system and play a crucial role in the detection of the movements of the checkers. The use of multiple PCB boards, along with hall sensors, resistors, and shift registers, makes this project a prime example of how advanced technology can be used to create innovative applications in the field of IoT.

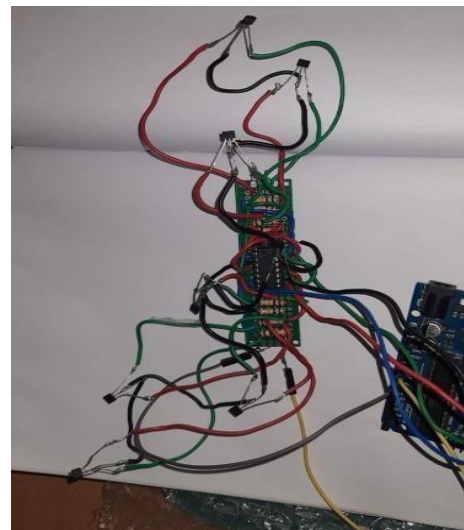


Fig -7: 8 hall sensors assembly

4.2.4 PCB Board Assembly

In this project, a total of 4 PCB boards are connected to each other in order to achieve the desired results. Each PCB board contains a number of hall sensors, resistors, and shift registers that work together to detect the movements of the checkers. The hall sensors are responsible for detecting the magnetic fields generated by the checkers, while the resistors are used to regulate the flow of electric current in the circuit. The shift registers play a crucial role in controlling the hall sensors and ensuring their optimal functioning.

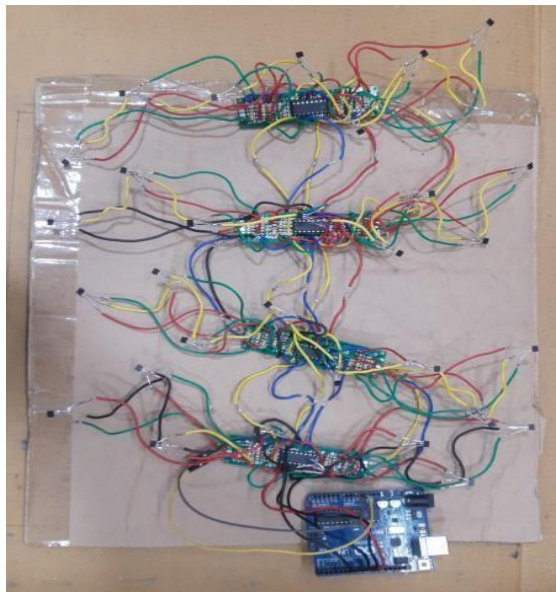


Fig -8: 4 PCB board connected.

5. FLOW CHART

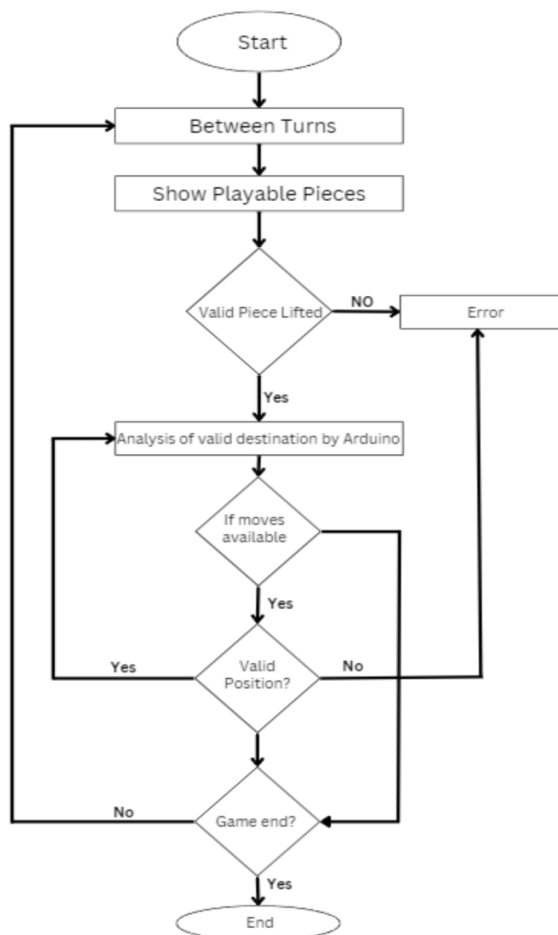


Fig -9: Flow Chart

The flow chart of checkers using IoT described as first the condition is at initial position. Going with the arrow the condition is checked between the turn and after that the direction goes to show the playable pieces on the board. If the valid piece is lifted, then the game continues if not then the system throws an error. The moves then analysed by the Arduino as it is valid or not and for that valid piece the moves are available on board or not if yes then the games continue in loop if not the board shows an error. After that the condition is checked that if the pieces or moves are available for any piece or not on board, if there is no moves or piece left then game will end but if not, the game again start from previous state. The moves are then evaluated by the Arduino to see whether they are legitimate or not, and whether they are available on the board for that valid piece or not. If they are, the game loops forward; otherwise, the board displays an error.

6. CIRCUIT DESIGN

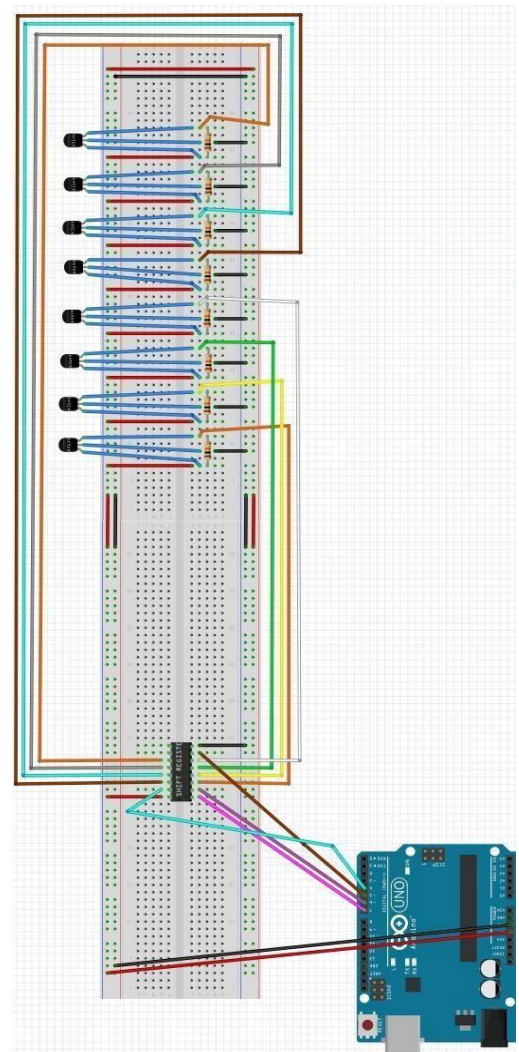


Fig -10: Circuit Design

The circuit is designed with the 8 hall sensors, 1 Arduino Uno, 1- 74HC165 8-bit shift register and 8 10K Ohm resistors. There is one clock pin, clock enable pin, load pin, data pin in shift registers and connected to the Arduino and there is parallel input and output. Pin 1,2,7, 15 are connected to the Arduino for controlling the 8 sensors via shift registers. Middle 8 pins of shift register upper 4 pins and lower 4 pins are connected to the 4 hall sensors of one side and other 4 pins are connected to the 4 hall sensors of other side. All the 8 hall sensors are connected to each 10K ohm resistors for controlling the amount of current in each sensor for proper working. And there are two other pins (1 and 2) of shift register that are connected to the Arduino for supply 5v current.

7. SYSTEM FLOW

7.1 System Flow

The system flow of the checkers game using IOT is a crucial aspect of this research. The first step in the system flow is to detect the initial positions of the pieces. This is done to keep track of the game and to ensure that each player's pieces are placed in their proper starting positions. The system then moves on to determining whose turn it is. This information is necessary to ensure that the players are aware of whose turn it is to make a move.

The system then highlights the pieces that can be moved by the player. This includes pieces that have a possible move and any opponent's pieces that can be cut by a piece of the current player. [5] If any opponent's piece is being cut by any piece of the current player, then the system will only highlight those pieces. On the other hand, if there is no opponent's piece being cut, the system will show the normal moves pieces.

Once the player lifts a piece, the system validates the chosen piece. If the chosen piece is one of the highlighted pieces, it will show all the possible next moves of that chosen piece. If the chosen piece is not a highlighted piece, the system will show an error message, indicating that the chosen piece cannot be moved.

After the player has selected a piece, they can move it to their desired location. The system then validates the move to ensure that it is a valid move. If the move is not validated, an error message will be thrown. If an opponent's piece is cut during the move, the turn will not change and it will keep on looping through the flow starting from, highlighting the possible next moves of the chosen piece. However, if an opponent's piece is not cut, the turn will change to the opponent, and the

system will loop back to highlighting the pieces that can be moved by the player.

In addition, the system flow of the Checkers Using IoT project is designed to detect the change of a piece into a King piece and to update the game board accordingly. This eliminates the need for manual intervention and ensures that the game is played in a fair and consistent manner.

The system flow of the player also includes the validation of all moves, including those made by the King piece, to ensure that the game is played according to the rules. In conclusion, the system flow of the checkers game using IOT is a complex process that involves multiple steps to ensure a fair and enjoyable game.

The system must keep track of the game and determine whose turn it is, highlight the pieces that can be moved, validate the chosen piece, and validate the move. The system must also detect if an opponent's piece has been cut, change the turn if necessary, and loop back to the beginning if the game continues.

7.2 Winning Condition

The winning condition of a player in the checkers game using IOT is a critical aspect of this research. The system flow must be designed to accurately determine the winner of the game. There are three possible ways for a player to win the game.

Firstly, if all the pieces of one player are cut, the opposite player wins. [2] This means that if a player has no pieces left on the board, the other player is declared the winner. This rule is straightforward and ensures that a player is declared the winner if they have successfully captured all of their opponent's pieces.

Secondly, a timer can be integrated into the game to add an extra layer of excitement. If the time is over, the player with more pieces on the board wins. This means that if a player has captured more of their opponent's pieces, they will be declared the winner, even if the other player has some pieces left. This rule adds an extra level of strategy to the game, as players must consider the number of pieces they have, and the amount of time left to make their moves.

Finally, if the number of pieces is the same for both players, the game is a draw [2]. This means that if both players have the same number of pieces left on the board, neither player wins, and the game ends in a draw. This rule ensures that if neither player has an advantage over the other, the game ends in a tie.

In conclusion, the winning condition of the checkers game using IoT is a complex and multi-faceted process that requires the integration of advanced technology and a well-designed system flow. The system must be able to accurately detect the movements of the checkers, store information about the game in real-time, and determine the winner based on the three possible outcomes. The integration of the King piece into the system flow of the Checkers game adds an extra layer of complexity to the system but is a critical component of the system that must be designed and integrated correctly in order to accurately determine the winner of the game and loser of the game with more precision.

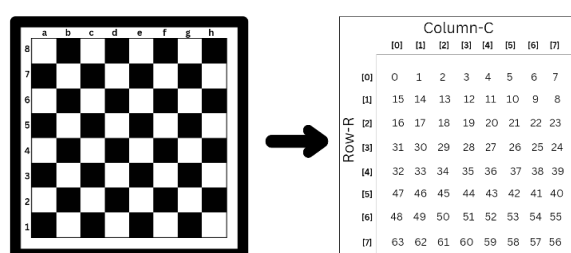


Fig -11: Checkerboard Position

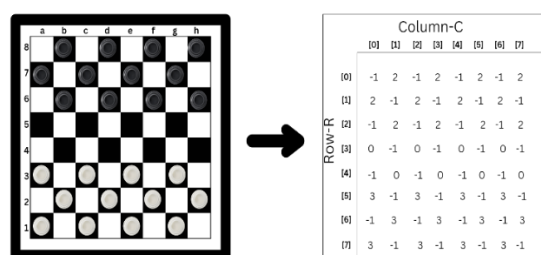


Fig -12: Board Position with Pieces





Representation of Board And Pieces		
	Values	Color
	2	Black Piece
	3	White Piece
	0	Black Square
	-1	White Square

Fig -13: Piece Value representation

8. CONCLUSION

In conclusion, the Checkers Using IoT project is a remarkable example of how technology can be used to enhance traditional games. The integration of IoT devices and systems provides an interactive, intelligent, and user-friendly experience for players. The system flow of the game has been designed to accurately detect the initial positions of the pieces, determine whose turn it is, highlight the pieces that can be moved, validate the chosen piece and its moves, and determine the winner of the game. The game's winning condition has been defined by three possible outcomes: if all the pieces of one player are cut, if the time is over and the player with more pieces on the board wins, or if the number of pieces is the same for both players, and the game is a draw.

The Checkers Using IoT project shows how IoT technology may be used to enhance conventional games and provide players a distinctive and interesting experience. It shows the significance of integrating IoT into numerous elements of our daily lives and opens up a world of possibilities for future developments in gaming technology. The development team's ingenuity and inventiveness are demonstrated by the project's successful completion, and the prospect of further developments in this area is intriguing.

9. FUTURE SCOPE

The future scope of the Checkers game using IoT presents exciting opportunities for innovation and growth. One area of focus is the integration of Artificial Intelligence (AI) into the game. The use of AI in Checkers will allow for a more challenging and dynamic experience for players. The AI algorithm can be trained to play at a high level, allowing players to compete against a challenging opponent. This will make the game more engaging and entertaining for players of all skill levels.

Another area of focus is the integration of remote play into the Checkers game using IoT. The game can be played both on a physical board and virtually, with one player playing on the board and the other playing remotely. This opens up new possibilities for players to enjoy the game, regardless of their physical location. With the integration of remote play, players can now compete against each other from anywhere in the world, making the game more accessible and convenient.

In addition to these exciting developments, the future scope of the Checkers game using IoT will likely include advancements in game graphics, sound effects, and

other sensory experiences that will make the game more immersive and engaging. The integration of new technologies, such as augmented and virtual reality, will further enhance the player experience and provide new opportunities for players to engage with the game in innovative ways. Overall, the future of the Checkers game using IoT is bright and offers a wealth of opportunities for growth and innovation. With the integration of AI, remote play, and new technologies, the Checkers game using IoT is poised to become one of the most exciting and dynamic games in the world.

REFERENCES

- [1] Research Paper on chess by Siti Zarina Mohd Muji, Mohd Helmy Abdul Wahab, Radzi Ambar-Embedded Computing System (EmbCoS) Research Focus, Group, Department of Computer Engineering, Faculty of Electrical and Electronic Engineering, 86400, Parit Raja, Batu Pahat, Johor, Malaysia.
- [2] Research Paper on Checker Is Solved by Jonathan Schaeffer, Neil Burch, Yngvi Björnsson, Akihiro Kishimoto, Martin Müller, Robert Lake, Paul Lu, Steve Sutphen, Department of Computing Science, University of Alberta, Edmonton, Alberta T6G 2E8, Canada.
- [3] Similar smart board game project from <https://www.bogdanberg.com/2017/04/21/smartgeboard-updates/>
- [4] Research Paper on Study of Artificial Intelligence into Checkers Game using HTML and JavaScript by K Idzham K1, W N Khalishah M1, Steven Y W1, M F Aminuddin MS1, N Syawani H1, Zain AM1, 2 and Y Yusoff1, 2 1 School of Computing, Faculty of Engineering, Universiti Teknologi Malaysia, 81310 UTM Skudai, Johor Bahru, Johor, Malaysia.
- [5] Research paper on Design and Implementation of a Wireless Remote Chess Playing Physical Platform by Divye Bhutani1, Yusuf Ali2, Pratyush Gupta3 Delhi Technological University, Shahbad Daulatpur, Delhi, India.
- [6] Research Paper on Solving the Game of Checkers Jonathan Schaeffer Robert Lake Department of Computing Science University of Alberta Edmonton, Alberta Canada T6G 2H1.