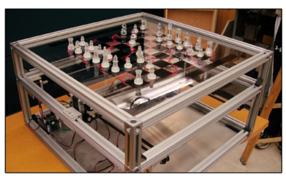
Automated Chess

Dec13-10: Riley Chapleau, Matt Dresser, Jordan Jacobson,
Riley Swindell, Allison Thongvanh, Blake Vermeer
Client & Advisor: Joseph Zambreno

Introduction

This project was developed to bring to life the Mechanical Turk--an 18th century mechanical illusion that allowed a chess master to hide inside a machine in order to appear to autonomously play a strong game of chess against a human. The client desired an intriguing and complex display for the department of Electrical and Computer Engineering. The main problem of interest is to automate chess moves determined by artificial intelligence (AI). Below is a photo of the Mechanical Turk next to our chess machine.



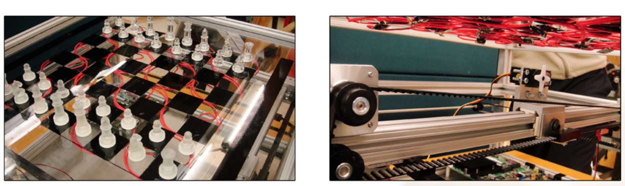
Design Requirements

Functional

- Piece movement system unobservable by players
- Movement system contained but able to be revealed
- Movement system must move fluidly, quickly, quietly
- User interface allows players to choose AI skill level
- User interface allows players to save and reset games
- User interface allows players to undo moves

Non-Functional

- Game base and pieces not overly large
- User interface comparable to computer chess games
- Autonomously moving pieces appear to move magically

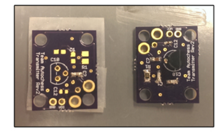


Details (Above): Detailed images of the playing surface and linear motion mechanisms.

Technical Details

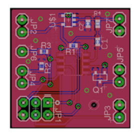
Power Transmitter PCB

The transmitter uses a Hartley Oscillator, oscillates at 2.6MHz, and wirelessly transmits power to the chess piece PCB.



Transmitter (Right): An unpopulated transmitter next to a populated transmitter

Power Management PCB

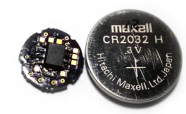


This PCB is used to allow a single button to turn on the chess board. It uses an ATTiny13 to control power cycling.

Power Management Layout (Left): The board layout for the power management board

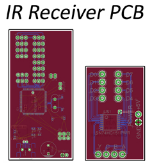
Chess Piece PCB

This PCB has an IR beacon to identify the chess pieces and is powered through magnetic induction. The beacon program runs on an ATTiny13 and IR codes are transmitted as serial data over IR.



Chess Piece PCB (Right): The chess piece PCB next to a watch battery.

IR Receiver PCB



This PCB receives and interprets IR codes sent by chess pieces. It also multiplexes between the IR receivers under each chess spot. It uses an ATMega128 to receive, interpret, and relay information about the chess piece's location and an FTDI chip to connect to the Atom board via USB

Receiver Layout (Above): Layouts of two modules of the IR Receiver board

Software Frontend

The part of the software allows the user to start a new game and choose the difficulty, and can be used for software debugging. It is programmed in C++ and uses the SFML library for graphics

User Interface (Right): Screenshot of part of the interface, viewing moves.

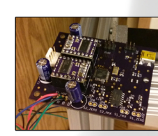
Software Backend

This part of the software uses the Stockfish chess engine to determine moves. This is also programmed in C++ and is where chess piece locations from the IR receiver PCB are received and movement commands are sent to the motion controller.

Stockfish (Left): The artificial intelligence chess engine used to determine moves.

Motion Controller PCB

This PCB controls the stepper motors and servo. It uses Pololu DRV8825 stepper motor drivers and four limit switches to determine bounds of stepper motor motion and zero position.



Motion Control (Right): The motion control board connected to the frame.

Design Approach

Our group approached the problem by splitting it into two modules: software and hardware. Sofware included the chess AI, infrared identification system, and user interface. Hardware included wireless power system and mechanical design.