Wifi Connected Chess Boards

Project Requirement Specification

Nick Kraus, Kyle Jameson, Maurice Wallace, Mark Mauriello

September 20, 2015

Purpose

The goal of our project is to create a physical implementation of an internet chess game. This will appeal to anyone who enjoys the ability to play chess at their will, but also prefers playing on a physical board. This project will allow those users to have the best of both sides of internet and board based chess.

Definitions

- Embedded System: A computer system that is within and controls a larger mechanical or electrical system.
- Microcontroller: A small computer with programmable inputs and outputs.
- UART: A Universal Asynchronous Receiver Transmitter, a method to serially send data over electrical wires.
- Reed Switch: An electrical switch which triggers in the presence of a magnetic field.
- Limit Switch: A switch that triggers when an object collides with it.
- Gantry: A structure with a moving platform.
- Stepper Motor: An electric motor which divides a full rotation into a number of discrete equal steps.
- API: An application programming interface, which allows different software systems to interface with each other.
- Client Server Architecture: An architecture in which the clients (which run the users application) communicate with each other through a central server.
- Bare Metal: A term referring to software that runs directly on a processor without any supporting operating system.

System Overview

Our project will have four main systems. These four systems are the electrical system, mechanical system, client software (firmware), and server software. The client software will be the firmware for the electrical system, and therefore control the mechanical system, as well as be the main application for the user. The server software will allow different clients to communicate with each other.

Electrical System

An embedded circuit board will house all of the electrical components, including the microcontrollers which will run the client software, and the WiFi module which will be used to communicate with the server software. The electrical system will measure the state of the chess board through reed switches, and control the mechanical system by using stepper motors and motor drivers.

Mechanical System

The mechanical system will control the movement of pieces on the chess board. It will use an X-Y gantry underneath the playing surface equipped with an electromagnet to move the pieces. The electrical system will be used to monitor and drive the mechanics to make sure they are operating properly.

Client Software

The client software is the bare metal firmware which runs the user facing application as well as controls all of the mechanical system. It consists of code for multiple microcontrollers and for the WiFi module, which all communicate with each other over a UART serial bus.

Server Software

The server software allows one chess board to communicate with another. It offers an API as an interface for the WiFi module to communicate with, and provides a central location for the chess boards to connect to.

Project Features

- Board State: Be able to read all chess positions to determine whether a piece is at that position or not. (Basic)
- Opponent Connection: Be able to connect to an opponent board in order initiate the start of a chess game. (Basic)
- Opponent Communication: Communicating the current state of one board to the other to advance the game a turn. (Basic)
- Move Recognition: Recognize moves and only allow valid chess moves to be sent to opponent. (Basic)
- State Interaction: Have the chess board react to an opponents game state by relocating pieces to their correct position. (Basic)
- Game Completion: Implement a way to signal that a game of chess has been completed. (Basic)
- Piece Graveyard: A dedicated area for each piece to go when taken out of the game. (Basic)
- Virtual Interface: Allow a computer program to play a game of chess against the physical board. (Desired)
- Checkmate Detection: Detect if a checkmate has occurred and signal end of game if so. (Desired)
- Stalemate Detection: Detect whether a stalemate occurred and signal end of game if so. (Desired)
- Game Resume: Continue a previously played chess game from a predetermined starting point. (Desired)
- State Recording: Record all states of a chess game as its played. (Desired)
- Chess Timers: Include move timer and different variations of how they're used. (Desired)
- Arbitrary Positioning: Be able to relocate any correct board set up to any other correct one. (Dream)
- Chess Variants: Be able to play chess variations with all rules implemented by the system. (Dream)
- AI: Server side AI for playing against an artificial opponent. (Dream)
- Game Playback: Play through all moves of a recorded chess game. (Dream)
- Standardized Recording: Record all game moves in a standardized chess move format. (Dream)

System Interfaces

User Interfaces

This project will have two separate parts to its user interface, the actual chess board itself and a LCD screen with tactile buttons. The chess board and pieces will allow the user to control the state of the chess game, just like in a physical version of the game; moving a piece will move that equivalent piece on the opponents board. The LCD and buttons will allow the user to stages and options within the game, such as marking the end of a turn or setting up a new game with another player.

Hardware Interfaces

The client software is firmware for the microcontrollers that make up the embedded system. These microcontrollers will be used to get user input and control the mechanical system. The microcontrollers have programmable inputs and outputs which will allow us to get the state of the chess board, control the stepper motors and electromagnet that will move the pieces, drive displays and button arrays for the user input, and communicate over a WiFi network.

Software Interfaces

This project will have two sets of software that will interface with each other: the client software and the server software. The client software will connect to the server through a TCP connection and use the API that the server exposes to set up games with other players and communicate the game state to the opponents board.

Communication Interfaces

Communication in this project is split up between communication in the embedded system and communication that connects the client to the server. To connect the client to the server a TCP connection will be used over a WiFi network. This connection will allow the client to reliably connect to the server, and make sure that no packets are lost during data transfers. The embedded system will be interconnected using UART buses. These will allow any microcontrollers to communicate with each other and allow the WiFi module to connect to the microcontrollers.

Dependencies

Either a web server or a software interface to an existing chess server will be required for this project to operate correctly. The platforms this will run on are yet to be determined, but there are many choices such as running a local server, using an existing server architecture like Amazon Web Services, or using the existing chess servers at freechess.org.