Software Overview

Year: 2020 Semester: Fall Team: 22 Project: Social Distancing Chess

Creation Date: 9/10/2020 Last Modified: September 10, 2020

Author: Fischer Bordwell Email: fbordwel@purdue.edu

Assignment Evaluation:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item** | **Score (0-5)** | **Weight** | **Points** | **Notes** |
| **Assignment-Specific Items** | | | | |
| **Software Overview** |  | x2 |  |  |
| **Description of Algorithms** |  | x2 |  |  |
| **Description of Data Structures** |  | x2 |  |  |
| **Program Flowcharts** |  | x3 |  |  |
| **State Machine Diagrams** |  | x3 |  |  |
| **Writing-Specific Items** | | | | |
| **Spelling and Grammar** |  | x2 |  |  |
| **Formatting and Citations** |  | x1 |  |  |
| **Figures and Graphs** |  | x2 |  |  |
| **Technical Writing Style** |  | x3 |  |  |
| **Total Score** |  | | |  |

5: Excellent 4: Good 3: Acceptable 2: Poor 1: Very Poor 0: Not attempted

General Comments:

*Relevant overall comments about the paper will be included here*

1.0 Software Overview

The software component of Social Distancing Chess consists of two aspects: the firmware operation of the board’s microcontroller and the smartphone application that interfaces with the user’s and opponent’s board states. These two devices will communicate by means of their Bluetooth receivers. Because of this, the board and the phone must be in proximity in order to function.

The STM32F0DISCOVERY kit that we will be using features 64 KB of Flash memory and 8 KB of RAM [1]. The STM32F051R8 microcontroller unit will be responsible for managing and manipulating the board state and peripheral LEDs. The GPIO pins will be used as buses that will be multiplexed to control the board LEDs, which will give information relating to moved pieces. Additionally, the microcontroller will interface with our Bluetooth module in order to sustain a wireless connection with the user’s smartphone.

Furthermore, the microcontroller will read the board state by means of the readings outputted by the Hall effect sensors throughout the board. These sensors will be placed on each square on the board and read the magnitude of the magnetic field on their position. This information will be relayed to the microcontroller which can be used to determine the board’s state. The purpose of this is so that users can move pieces as if they were playing a normal chess game without having to input their moves manually into an electronic device.

For Bluetooth communication, the HC-05 Bluetooth Module will be used in connection to the microcontroller. The HC-05 will be used in Data mode in which it can send and receive data from other Bluetooth devices. The HC-05 uses Serial Port Protocol (SPP) so it can read data from the microcontroller easily through its Tx and Rx pins [2]. The first step in the setup of the board will be pairing with the user’s smartphone, which will be prompted in the application.

The STM32 microcontroller will communicate with the user’s Android smartphone via SPI. The data relayed by the microcontroller to the phone will be sent to a server with an asynchronous HTTP request. The server will see both board states and handle sanity checks and chess logic. Amazon Web Services offers scalable server hosting which we will use for this software [3].

2.0 Description of Algorithms

The movement of a board piece will be determined by the readings relayed by the Hall effect sensors under the initial piece position and the final piece position. When a piece is lifted from the board, the LED under the piece will flash to indicate the initial position. The LEDs under possible positions to move to will also flash at this time. If a piece is moved to an illegal spot, data will not be relayed to the server, but rather the LED will flash red to indicate the piece must be moved back to its original spot. Because the possible moves of chess pieces are simple and well-defined, it is anticipated that this will require a low amount of computation and Flash memory.

After a move, the board state will be relayed to the web server. The web server will at this point check for check or checkmate scenarios. Numerous open source solutions exist for chess engines which we can utilize for this portion of the design [4].

3.0 Description of Data Structures

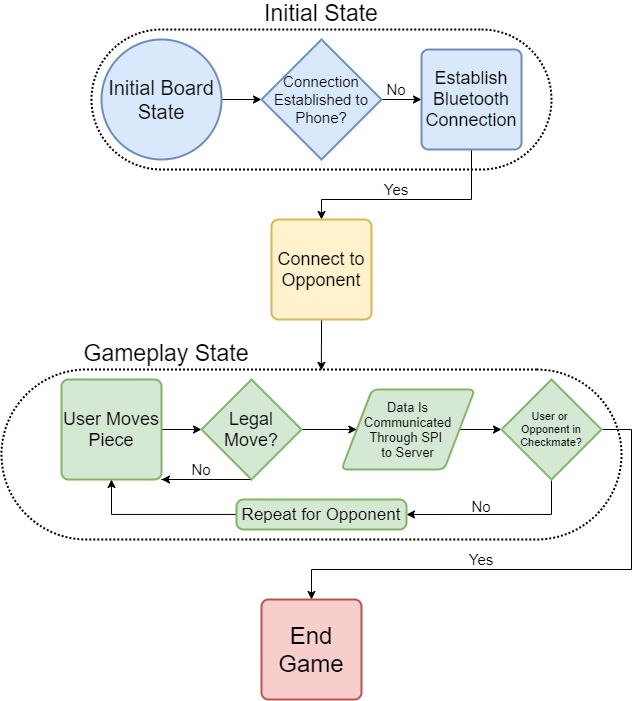
The most significant data structure handled by the microcontroller will be the board state of the user’s chess board. The board state will be stored as an array of structs. An individual struct in this array will represent one piece on the board and consist of the following members: x position, y position, and piece type. This is all the information required to fully represent the board and moves in chess. This will be relayed through SPI to the Bluetooth module, which will in turn communicate the information to the user’s phone and then to the web server.

The chess engine used by the web server will be more complicated. When a legal move is read by the server, the server will determine whether the king is in check, as checkmate is how our win condition is determined. At this point, all possible moves will be analyzed to determine if one will lead to a non-check situation. If there are no such moves, the game will end and the appropriate information will be sent to the user’s and opponent’s boards and displayed on the LED screen.

4.0 Sources Cited:

1. st.com. *Discovery kit with STM32F051R8 MCU.* Accessed on: Sep. 10, 2020. [Online]. Available: <https://www.st.com/en/evaluation-tools/stm32f0discovery.html>
2. components101.com. *HC-05 - Bluetooth Module.* Accessed on: Sep. 10, 2020. [Online]. Available: <https://components101.com/wireless/hc-05-bluetooth-module>
3. aws.amazon.com. *Application Hosting Using AWS*. Accessed on: Sep. 10, 2020. [Online]. Available: <https://aws.amazon.com/application-hosting/>
4. *Sunfish.* Accessed on: Sep. 10, 2020. [Online]. Available: <https://github.com/thomasahle/sunfish>

Appendix 1: Program Flowcharts

**

Appendix 2: State Machine Diagrams

