**AUTONOMOUS   
 CHESS BOT**

Project- ROBOREX\_CHESS\_MATE

by Team ROBOREX

Team Members:

* Neelam Mahapatro
* Prabin Kumar Rath
* Swostik Acharya
* Subhum Sahoo
* Pabitra Dash
* Nischay Nirajit Mahanta
* Ankit Behera

**Goal of the project: -**

The goal of this project was to create a physical chess board that is able to interact with a human player and a computer. The user’s move is sent to the system through image processing. Stockfish, a chess engine sends its move through the computer and the board moves the stockfish piece using a mechanical XY positioner with an attached electromagnet.

**Mechanical Description: -**

The main part of the mechanical design is the x-y slider working to move the chess dots around. The slider mechanism has been designed using two sliders fixed in x-y direction. Two stepper motors have been used to move the slider. There are pinions attached to the stepper motors and they run on rack made of plastic and are fixed by the side of the sliders.

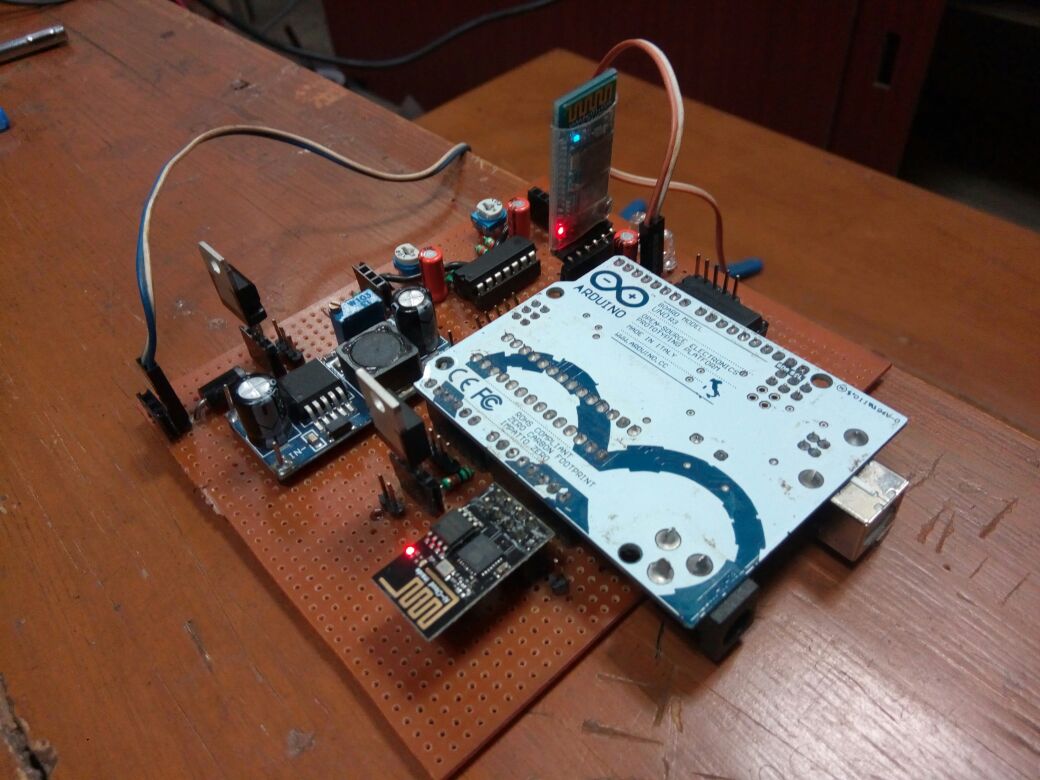
Arduino UNO has been used to control the stepper motors with an embedded code running on the background and the motor moves in steps as received. The whole setup has been placed on a ply board. The x railing has been placed on another piece of ply. There is a block to hold the electromagnet. The whole system has been so designed to overcome any positional error that may surface therein.

**Electronics Arena: -**

A 12-volt supply is given to the buck converter, Arduino UNO, LM 7805 voltage divider and an electromagnet. The Buck converter converts the 12-voltage input to high current 5 volts output that is required for the stepper motors to work. Electromagnet is directly supplied with 12 volts as it consumes relatively more amount of current and a N-MOS is used to switch this electromagnet.

Another essential part is IR Feedback system whose function is to avoid the mechanical error generated while positioning the electromagnet as per the move produced by stockfish for a certain chess piece. We have used LM 324, a quad opamp IC for our feedback system. This system inspects whether the magnet had moved to its specified position or not.

Arduino is a 16MHz microcontroller which is used to handle the hardware unit of the bot. The entire circuit is soldered on a vero board and mounted on the X slider.



**Software Description: -**

Especially we have used the ROS (Robot Operating System) framework in our project. It helped us to develop critical and robust software for our robot. Suppose a node is written in a programming language cpp and requires the output of another node to be executed. The second node may be written in same programming language or different programming language like java, python. So, here the message from one node is transferred to the other node by a default or customised message file through the ROS. The nodes we used here are chess ai, board handler, frame\_pub and frame handler user interface i.e. UI 1(display) and UI 2(interactive), Arduino.

The game starts from the UI 2 node. In this UI, the turn or the player is represented by a dot. User has an opportunity whether he would like to start the game or let stockfish to start. If user wishes to start the game, then he physically does the change on the chess board after entering **yes**.

The movement accomplished by the user is detected by camera which captures the image of the whole chess board including the pieces. Frame\_pub node accesses that camera image using OpenCV and encodes it to a sensor message and forwards to the frame handler node through ROS. Frame handler node detects all the positions where the chess pieces are present on the board and encodes it into a binary string format. This binary string is transferred to the board handler node through a message file. The board handler node compares the two subsequent binary strings sent by the frame\_handler node to generate position changes in an encoded string format.

Chess\_ai node gets the string from board handler node and converts it into UCI form. UCI is Universal Chess Interface which is a protocol for communicating with the chess engines, here to communicate with Stockfish. If the move is legal then in the next step the move is pushed to an ASCII matrix. ASCII matrix is internally stored from which Stockfish observes the board’s configuration to give a best possible move according to it. The move given by the STOCKFISH is updated in the ASCII matrix.

The current FEN (Forsyth-Edwards Notation) for the board is frequently passed to the UI 1 for display. Basically UI 2 node interacts with the user. From the UI nodes we are able to realise that the system is really working and playing with the user. Now, the output move given by the system is performed on the chess board autonomously through the mechanical unit present below the board.

