Department of Computer Science & Engineering
The University of Texas at Arlington

Detailed Design Specification
CSE 4317: Senior Design II
Spring 2023

Team CheckUR5
Checkers-Playing UR5 Co-bot

Nimita Uprety
Patricia Rojas
Hoang Ho
Kevin Vu
Joanna Huynh

Revision History

Revision	Date	Author(s)	Description
0.1	2.01.2023	HH	document creation
0.2	2.24.2023	HH, KV, JH, PR, NU	complete draft
0.3	2.24.2023	HH, KV, JH, PR, NU	release candidate 1
0.4	5.10.2023	HH, KV, JH, PR, NU	final draft

CONTENTS

List of Figures

List of Tables

1 Introduction

The UR5 co-bot will be programmed to be able to play the interactive & strategy-based game, checkers, against a human opponent. The robot will have an electropermanent magnetic attachment that will allow it to pick up and move the individual checkers pieces. The robot is not intended to beat the human opponent every time but rather just have the ability to play a full game of checkers against them. Our project aims to showcase the UR5 co-bot's abilities as a means of promoting the UT Arlington College of Engineering to prospective students through the means of an interactive demonstration.

2 System Overview

The system for our product is currently simplified into 4 layers and 3 high-level components. Each layer/component will interact with one another in different ways.

The Opponent is the person that will be going against the robot during the checkers game. The Input Device is one of the ways the opponent will be able to facilitate the progression of the checkers match. The camera is the component that acts as the eyes into the real world for our system. The Move Decision layer is the layer that will make all of the move decisions that the robot will make. This layer consists of Computer Vision and AI. The Computer layer is the central piece of our system. This layer acts as the facilitator of the checkers match, and receives input from the Opponent to progress the match, sends signals to the AI layer to grab board state and make a move, and signals the UR5 Robot Arm layer to execute those moves. The Move Execution layer is the layer that will physically execute the moves that our system will make during a checkers match with the robot arm and related parts. And finally, The External Components layer is composed of all of the physical checkers board game pieces. Both the Opponent component and the Move Execution layer physically interact with the External Components layer when they have made their move decisions to progress the match.

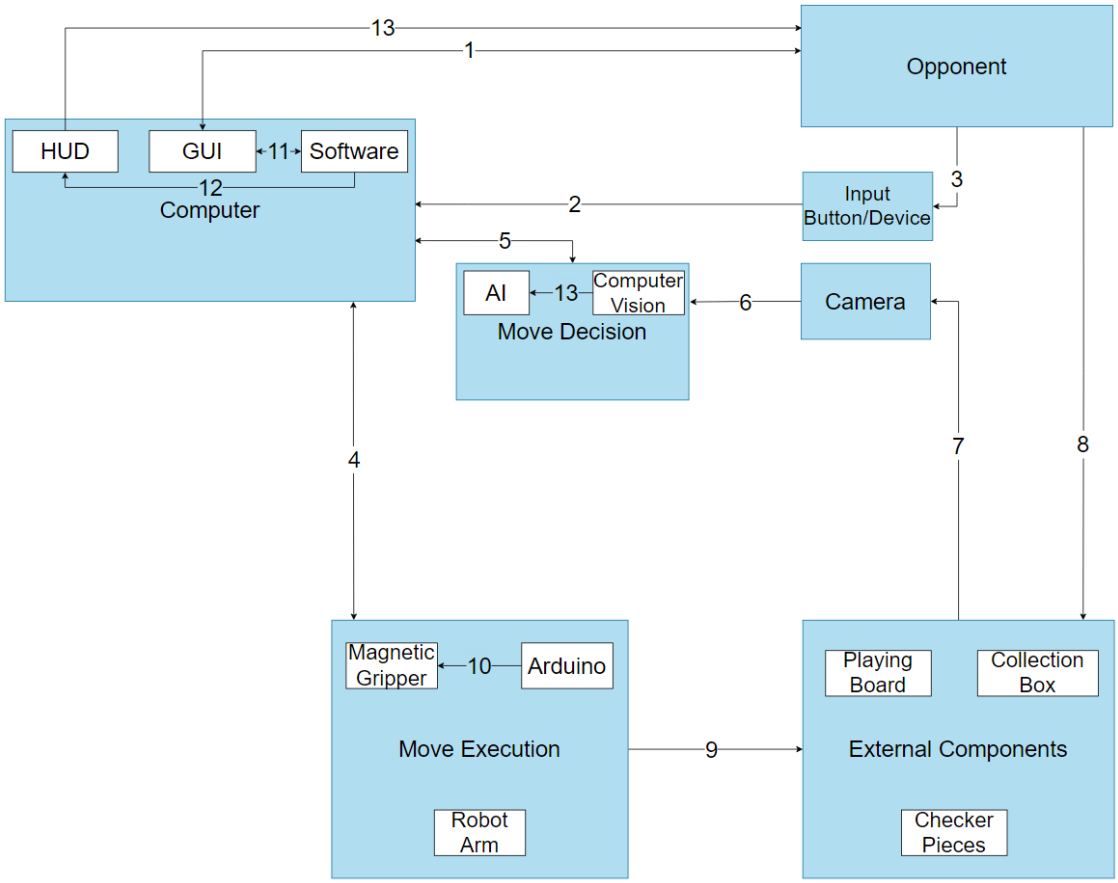

Figure 1: System architecture

3 Computer Layer Subsystems

In this section, the Computer layer is described in detail. This layer consists of the HUD, GUI, and Software subsystems. Overall, the layer is responsible for communicating with the players information about the state of the game, sending the robot arm the actions it needs to take, and handling the decisions after each move.

3.1 Layer Hardware

The major hardware component of this layer is the computer itself. This computer is communicating to the UR5 robot arm through an Ethernet cable connected to the same router as the Ethernet cable connected to the UR5 control box. Additionally, as part of the computer, a monitor to display information to the player. A USB is also required to send to the Arduino the source code written to control the magnetic gripper.

3.2 Layer Operating System

The layer requires Linux distribution Ubuntu 22.04 LTS.

3.3 Layer Software Dependencies

The layer will use the Python 3.10.6 programming language. It will also need to use the Arduino programming language to write the code to be executed on the Arduino that controls the magnetic gripper. Additionally, it is dependent on the Move Decision layer.

3.4 HUD Subsystem

The Heads-Up Display (HUD) subsystem is a class that communicates with the Software subsystem of the Computer layer to receive information about the status of the game.The HUD subsystem displays to the user the status of the game and any information needed for the user. In this case the user is the Opponent layer. Information on the status of the game includes the current scores and a display of the game board on the computer monitor. This information is provided by the Software subsystem. Each time a player overtakes another player's checker piece, the score will be updated. When the physical game board changes, such as when a player moves a piece, this will also be reflected on the screen through the HUD.

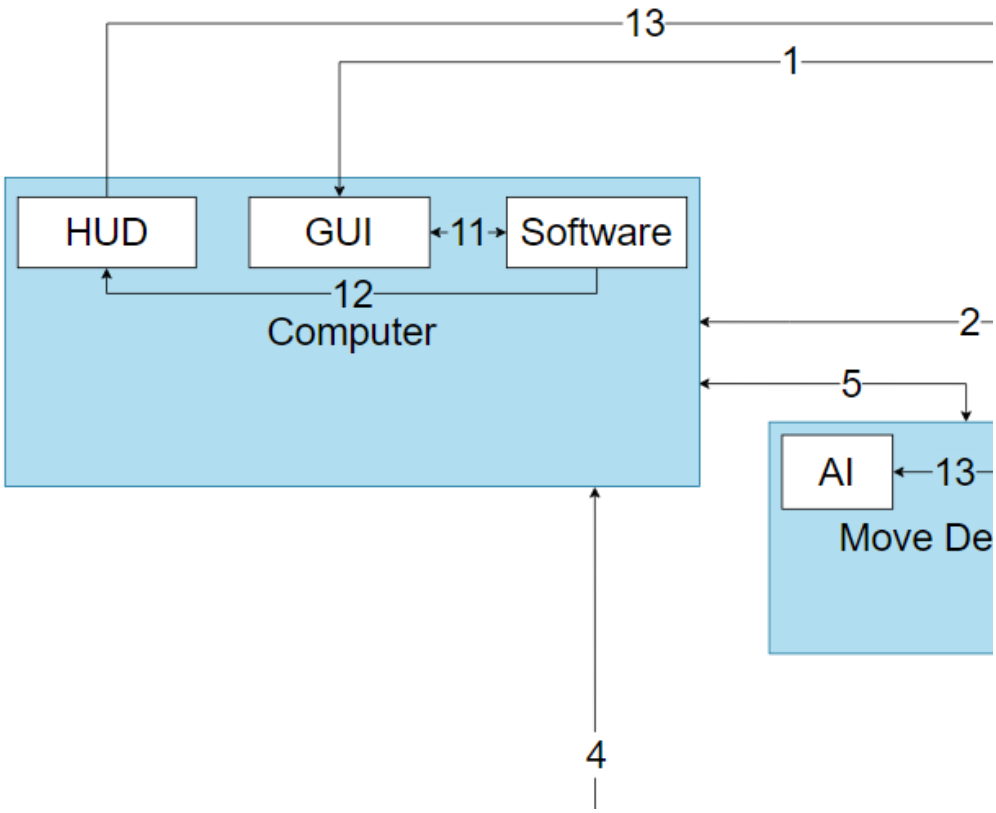

Figure 2: Computer layer subsystems diagram

3.4.1 Subsystem Hardware

A monitor connected to the computer to display game state status to the player.

3.4.2 Subsystem Software Dependencies

This subsystem is dependent on the Software subsystem to receive information about the status of the game to display. It will also have software dependencies on GUI libraries such as PyQt5 or Tkinter.

3.4.3 Subsystem Programming Languages

The subsystem uses Python 3.10.6 as the programming language used.

3.4.4 Subsystem Data Structures

N/A

3.4.5 Subsystem Data Processing

N/A

3.5 GUI Subsystem

The Graphical User Interface (GUI) subsystem is a class that is the main interactive component between the Computer and Opponent layers. It communicates with the Software subsystem within the Computer layer for its inputs and outputs. The GUI displays buttons for the user (Opponent) to interact with. These buttons will be able to change the state of the game for purposes such as new game, restart game, and quit game. Each time the Opponent interacts with the GUI by clicking a corresponding button to a function, that button will send a mouse click signal for that button to the Software subsystem. This signals to the Software subsystem to execute the function associated with the button that is clicked. For example, if the user clicks a new game button, the software executes the processes needed to reset the state of the game and UR5 robot arm.

3.5.1 Subsystem Hardware

A monitor connected to the computer to display the different options to the player. A mouse is also required for the player to interact with the GUI.

3.5.2 Subsystem Software Dependencies

This subsystem will have software dependencies on GUI libraries such as PyQt5 or Tkinter.

3.5.3 Subsystem Programming Languages

The subsystem uses Python 3.10.6 as the programming language used.

3.5.4 Subsystem Data Structures

N/A

3.5.5 Subsystem Data Processing

N/A

3.6 Software Subsystem

The Software subsystem is the main program for managing the checkers game itself and is behind the actions the UR5 Robot Arm will execute. This subsystem not only includes the source code itself, but also software packages needed for the program. The Software subsystem is responsible for handling events from the GUI. The subsystem also communicates with the UR5 robot arm through the Computer layer. The subsystem handles data on the status of the robot arm received from the robot arm itself. It also communicates to the robot arm what programmed actions it will need to perform. These actions will be determined by its interactions with the Move Decision layer which consists of the AI subsystem. The Software subsystem is also responsible for handling inputs from the Input Button/Device layer through the Computer layer.

3.6.1 Subsystem Hardware

N/A

3.6.2 Subsystem Software Dependencies

The Software subsystem requires the ur_rtde interface library to communicate with the UR5 robot arm. This library will allow us to write Python code to receive data from the UR5, and also control it. This is done using the RTDE Receive Interface API and RTDE Control Interface API from the library respectively. Documentation for the library can be found here: https://sdurobotics.gitlab.io/ur_rtde/index.html.

Controlling the UR5 includes communicating with the Move Execution layer and sending the commands to move the robot arm to a certain position. The subsystem is dependent on the Computer Vision and AI subsystems from the Move Decision layer to determine the commands it needs to send to the robot arm. It also needs this dependency to manage the checkers game and send the data required by the HUD subsystem. Moreover, it is also dependent on the GUI subsystem and Input Button/Device layer to listen to events from the player and handle them for the checkers game accordingly.

3.6.3 Subsystem Programming Languages

The subsystem uses Python 3.10.6 as the programming language used.

3.6.4 Subsystem Data Structures

N/A

3.6.5 Subsystem Data Processing

N/A

4 Move Execution Layer Subsystems

In this section, we describe the move execution layer. This layer is responsible for one of the key functionalities of our project which is the UR5 Robotic Arm's movement, which allows for the checkers game to take place between the user and the co-bot. The move execution layer is made up of three subsystems, the UR5 Robot Arm itself, the electromagnetic gripper, as well as an Arduino.

4.1 Layer Hardware

In terms of the Move Execution Layer the most heavily involved hardware component for this layer is the UR5 robotic arm itself.

4.2 Layer Operating System

This layer utilizes Robot Operating System (ROS), although we do not primarily use ROS as we are primarily using Python to move our co-bot. ROS will be utilized to check the status of our robot's arm movements/coordinates at any given time.

4.3 Layer Software Dependencies

This layer will also utilize the Python 3.10.6 programming language.

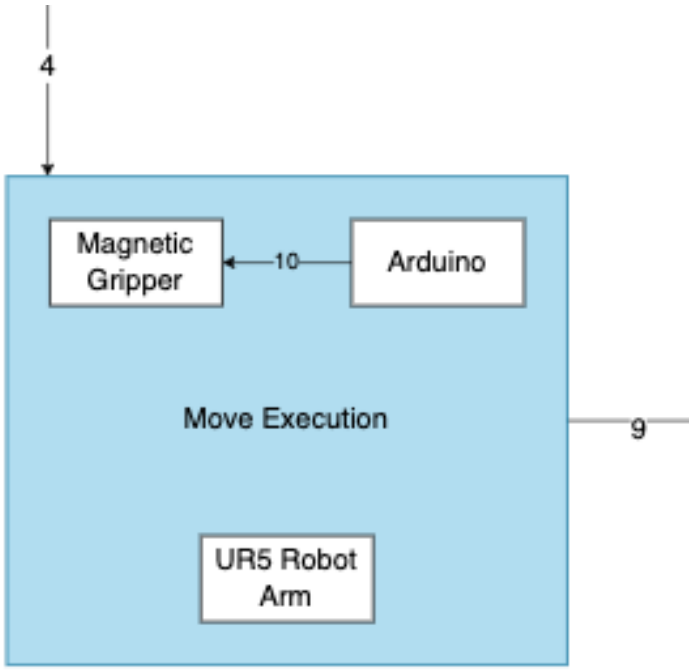

Figure 3: Move Execution subsystem description diagram

4.4 UR5 Robot Arm

The UR5 co-bot arm must be able to follow the instructions as asked by the software in order to hover over the correct location on the checker board. The robot must be able to move to fixed locations as predetermined by ROS and our custom software in order to not only move to specified pieces on the board but also to the collection box, and a waiting position.

The UR5 co-bot arm is responsible for the main mechanics of our project. As its ability to move to move to a spot on the board and pick up a piece and move it to another position is necessary so that the

UR5 can play against its human oppponent.

4.4.1 Subsystem Hardware

In the UR5 Subsystem as well as the Move Execution Layer, the hardware component involved is the UR5 robotic arm itself.

4.4.2 Subsystem Operating System

The operating system required here is ROS.

4.4.3 Subsystem Software Dependencies

This subsystem layer will require a Universal Robots Python library so that we can control the co-bots movements through python scripts. The Python library we intend on using for this is ur_rtde.

4.4.4 Subsystem Programming Languages

The UR5 subsystem requires the Python 3.10.6 programming language.

4.4.5 Subsystem Data Structures

The pre-recorded coordinates of our robot's movements will be saved into a multi-dimensional array so that our robot will have a list of saved positions to go to. The decision on where to go is determined by our Move Decision layer.

4.4.6 Subsystem Data Processing

N/A

4.5 Magnetic Gripper

The Magenetic subsystem of our Move Execution Layer is responsible for the UR5 robotic arm's ability to "pick-up" each checkers piece. This subsystem is integral to the project as a whole because without it the pieces will not be moved and the checkers game cannot be played.

4.5.1 Subsystem Hardware

This subsystem requires more hardware components than some of our other subsystems. It consists of an Arduino, an electromagnet, transistor NPN(2N3904), diode(1N4148), 22K Î© resistor, wires, breadboard, as well as a 3D printed attachement so that it can be screwed into the UR5 arm.

4.5.2 Subsystem Operating System

N/A

4.5.3 Subsystem Software Dependencies

This subsystem layer requires the Arduino 2.0.3 IDE. As that is needed for us to provide python scripts to Arduino which in turn will power our electromagnet.

4.5.4 Subsystem Programming Languages

The UR5 subsystem requires the Python 3.10.6 programming language.
N/A

4.5.5 Subsystem Data Processing

N/A

5 Move Decision Layer Subsystems

In this section, we describe the move decision layer in detail. This layer consists of Artificial Intelligence and Computer Vision. Overall, this layer is responsible for one of the most important parts of this project that is the decision making process of the UR5 robot for the Checkers game. This does so by determining what commands to send to the robot arm.

5.1 Layer Hardware

This layer requires the computer to utilize artificial intelligence and computer vision aspects of the decision making.

5.2 Layer Operating System

The layer requires Linux distribution Ubuntu 22.04 LTS.

5.3 Layer Software Dependencies

The layer will use the Python 3.10.6 programming language.

5.4 Artificial Intelligence

The Artificial Intelligence communicates with the camera and Computer layer subsystems to make decisions for the UR5 robot arm.

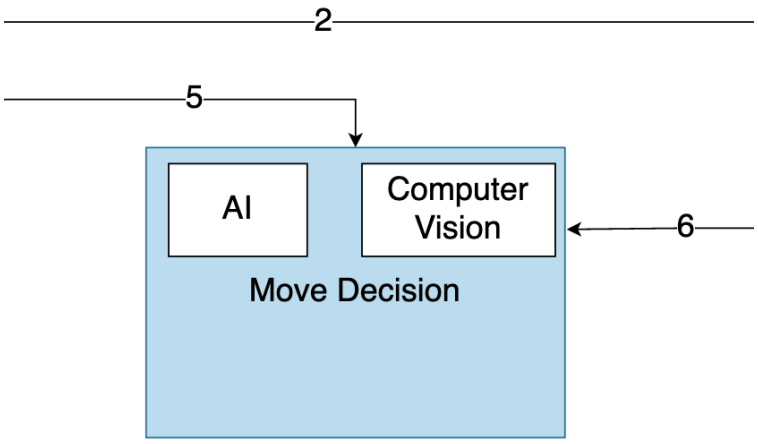

Figure 4: Example subsystem description diagram

5.4.1 Subsystem Hardware

This subsystem requires the computer to utilize artificial intelligence aspect of the decision making.

5.4.2 Subsystem Operating System

The subsystem requires Linux distribution Ubuntu 22.04 LTS.

5.4.3 Subsystem Software Dependencies

The subsystem will use python libraries designed for artificial intelligence.

5.4.4 Subsystem Programming Languages

This subsystem will use the Python 3.10.6 programming language.

5.4.5 Subsystem Data Structures

N/A

5.4.6 Subsystem Data Processing

N/A

5.5 Computer Vision

The Computer Vision interacts with the frames received from the camera to determine the player's move.

5.5.1 Subsystem Hardware

The subsystem will be using a camera to receive frames of the checkers position for both the players and will also use the computer for the computer vision process.

5.5.2 Subsystem Operating System

N/A

5.5.3 Subsystem Software Dependencies

This subsystem will use OpenCV and NumPy for the software dependencies.

5.5.4 Subsystem Programming Languages

This subsystem will use the Python 3.10.6 programming language.

5.5.5 Subsystem Data Structures

N/A

5.5.6 Subsystem Data Processing

N/A

6 External Components Layer Subsystems

In this section, the External Components layer is described in detail. This layer is comprised of the physical parts that make the checkers game, this includes the checkers playing board, the checkers pieces, and the collection box.

6.1 Layer Hardware

N/A or The components of this layer do now require any computer system hardware but it deals with the hardware of the checkers game, including the playing board, the checkers pieces, and the collection box.

6.2 Layer Operating System

N/A

6.3 Layer Software Dependencies

N/A

6.4 Playing Board

The Playing Board is the physical wooden platform with distinct cells which the checkers game is played upon.

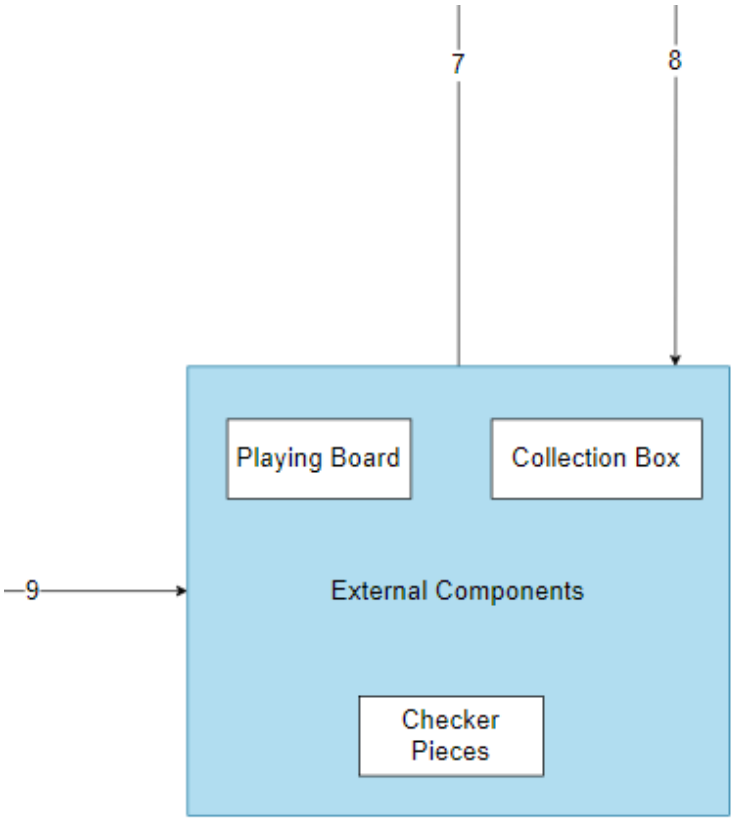

Figure 5: External components subsystem description diagram

6.4.1 Subsystem Hardware

N/A

6.4.2 Subsystem Operating System

N/A

6.4.3 Subsystem Software Dependencies

N/A

6.4.4 Subsystem Programming Languages

N/A

6.4.5 Subsystem Data Structures

N/A

6.4.6 Subsystem Data Processing

N/A

6.5 Checkers Pieces

The checkers pieces are the focus of the two players as they are the items that will be manipulated during the match.

6.5.1 Subsystem Hardware

N/A

6.5.2 Subsystem Operating System

N/A

6.5.3 Subsystem Software Dependencies

N/A

6.5.4 Subsystem Programming Languages

N/A

6.5.5 Subsystem Data Structures

N/A

6.5.6 Subsystem Data Processing

N/A

6.6 Collection Box

The collection box stands off to the side ready to hold the checkers pieces.

6.6.1 Subsystem Hardware

N/A

6.6.2 Subsystem Operating System

N/A

6.6.3 Subsystem Software Dependencies

N/A

6.6.4 Subsystem Programming Languages

N/A

6.6.5 Subsystem Data Structures

N/A

6.6.6 Subsystem Data Processing

N/A

7 Appendix A

7.1 CAD Models

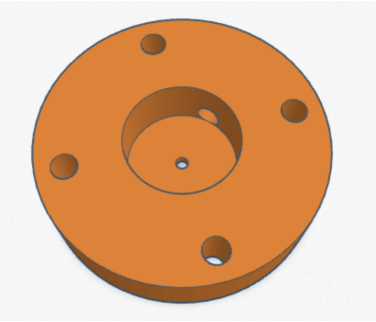

Figure 6: CAD Model of our Magnetic Gripper Attachment

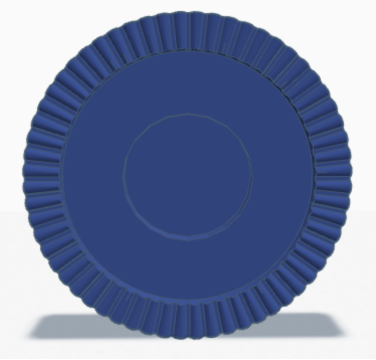

Figure 7: CAD Model of our Regular Checkers Pieces

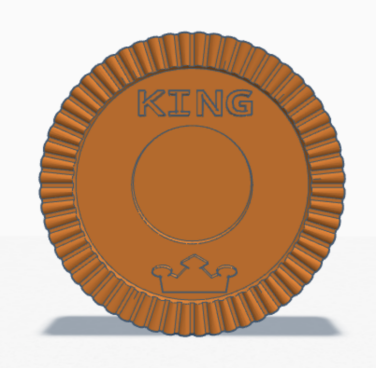

Figure 8: CAD Model of our King Checkers Pieces

7.2 Circuit Schematics

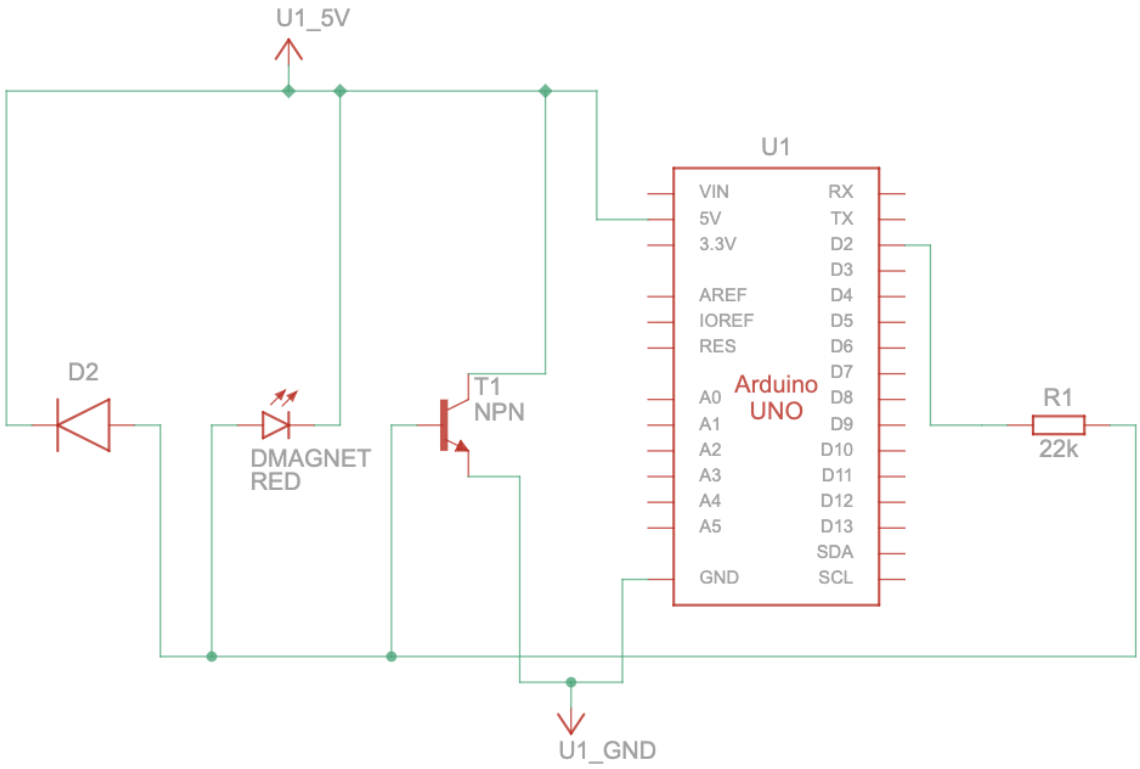

Figure 9: Schematic View of our Arduino & Electromagnet setup

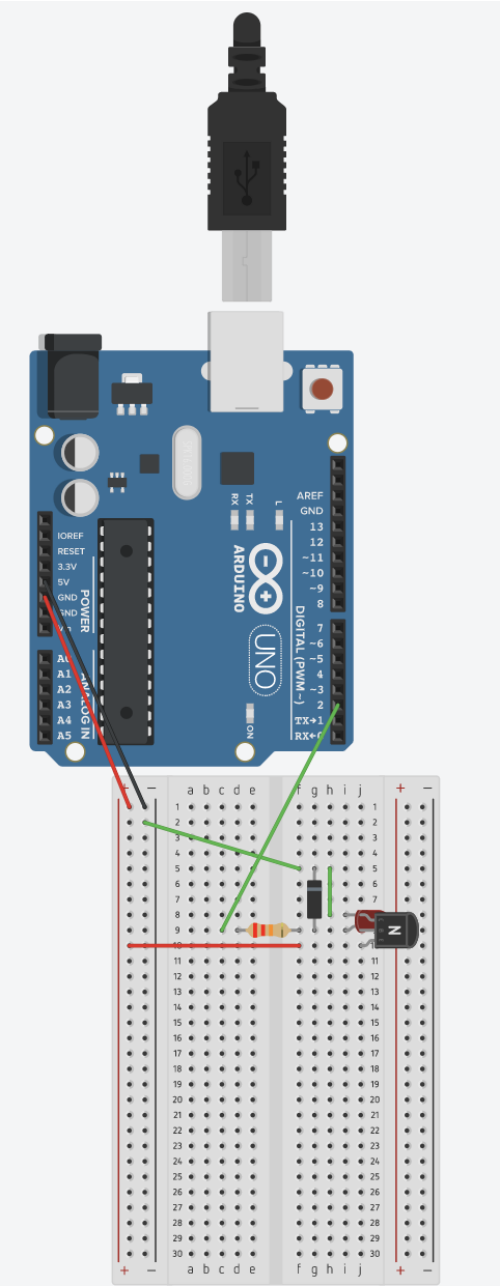

Figure 10: Circuit View of our Arduino & Electromagnet setup