1. Introduction

I have been working on the User Interface, and Control subsystems for both the board and the remote. The Control Subsystem for the board and remote are in charge of sending and receiving data back and forth. The User Interface Subsystem is in charge of taking user input and displaying data for the user. The user uses the remote system to control the board system. The control is limited to the speed and direction of the board. Within the board system, the boards control subsystem is in charge of receiving data sent over from the remotes control subsystem and behave accordingly.

1. Design

2.1 Wireless Communication

For wireless communications I decided to go with the NRF24L01 module which is a transceiver module allowing both systems to send and receive data. The very first circuit built was to ensure that we can have a stable 1 way communication. The built circuit can be seen in Figure XX and Figure XX. Here I built almost 2 identical circuits. What was different about the was the code that was uploaded to them. One of the circuits was always sending data and one was always receiving. This was simply a stepping stone toward building a 2 way communication for our project. To test 1-way communication working I decided to send a value based on the state of the button. If the button was pressed, the value sent would be 0.00, otherwise the value sent would be 1.00. The 1-way communication can be seen in figures XX and XX.

After getting 1 way communication working, I started with the first version of 2-way communication. The first approach was to constantly switch between sending and receiving modes. While this method worked, there was a slight amount of delay noticed during testing. Furthermore, it was an inefficient way. Thus, I switched so both systems are listening for packets. One board would send data every 50ms and the other would send every 55ms. Figure XX would show a snippet of code that shows that. While this was working, and delay seemed gone, it was still not the best way.



Finally, the method I am using now is a bit more complex than the previous, however, it works much smoother and allows for a bit more versatility and functionality. Now, the remote sends data to the board, the board acknowledges that the packet was received and within that acknowledgment message sends some data back. Not only this eliminates any issue by manually switching between sending and receiving data, it also allows us to detect disconnection of either system. Furthermore, it simplifies the packet sent and packet received by decreasing the amount of data within. In Figure XX the code that is in charge of sending and confirming the data was sent can be seen.

Text

Description automatically generated

2.2 Remote System

The remote system consists of power subsystem, user interface subsystem, and control subsystem. I worked on user interface and control subsystem.

2.2.1 User Interface Subsystem

Initially, I was planning on using the Wii Nunchuck as the remote for the user. The controller had a joy stick and a few buttons which could be used for user input. Thus, the circuit seen in Figure 1 was created on the bread board to read the inputs from the nunchuck. While this method was fully functional, as seen in Figure 2, a lot of the features such as the accelerometer, the second button, and the range of the joystick were not needed. Also, the amount of space within the controller would have made it difficult to fit in all the components. Also, taking into account that we would have been using another library to read the values of the nunchuck we would have been adding a bit of complexity to our code and might have increased a bit of delay. Thus, another group member decided to find an alternative for the housing of the controller, while I worked on developing the communication for the remote.

Therefore, the circuits showed in Figure 3 and 4 were created. One

2.2.2 Control Subsystem

2.3 Board System

2.3.1 Control Subsystem

1. Verification
2. Conclusion
3. Citation