**California University of PA**

**Dept. of Computer Science, Info Systems, and Engineering Technology**

**CET360 Microprocessor Engineering**

**Spring 2022**

**= Project Report =**

**Final Project: Autonomous Vehicle**

**Member 1: Andrew D. Bissell**

**Member 2: Noah Proctor**

**Date Submitted: 04/29/2022**

**Final Project: Autonomous Vehicle**

**I. OBJECTIVES**

Using the code examples, theory, components, and lessons for the CET 360 Microprocessor Engineering class, show an understanding of the course objectives. Be able to implement hardware and code with multiple different applications taught throughout the period of the course.

**II. PROCEDURE**

1. **Construction**
2. **General layout**

First see V. B. Table 1 for a complete list of components and V. B. Table 2 for tools useful for the project. With a built chassis, the construction can take on a multiple of forms. The one shown is not the best for every situation. First thing would be to find the parameters of the bots working environment and change the layout to the one best suited for the task. This construction narrows the chassis, so the tracks are mostly under the metal mounting plate. Using Lexan, a second platform can be built to add space for wiring and other components. Cut Lexan to about the same size as the black plate, the 3/16” Lexan can be scored and snapped (CAUTION: Snapping the Lexan can cause sharp edges make sure to snap using a rubber mallet or a thick gloved hand). The Tiva Board is situated on its side so that all the male and female pinouts can be reached, the micro USB side of the Board will have the L298N Motor Driver near it. The two 400-pin breadboards should be situated close to the power sources and grounds, they will also need to be grounded together. The 3.3V board will receive its 3.3V from the Tiva Boards 3.3V output pin, and the 5V board will receive its 5V from the L298N Motor Driver 5V output (NOTE: Make sure the jumper on the L298N Motor Driver is on so the 5V will be supplied). The Ultrasonic Sensors should be positioned in the center of each side of the bot either on the Lexan or the metal plate, make sure to leave room for the IR proxy sensors. The IR proxy sensors should be mounted on the metal plate as close to the chassis as possible, the front and rear sensors have mounting space across from the DC motor. See Section V. A. Figure 1 and 2 for diagrams for the layout.

1. **Specific Modifications**

This section will cover specific modifications for the autonomous vehicle. The micro-USB cord used to power and transfer data to the Tiva board, will be cut with enough length to reach the Tiva Board and the L298N 5V and Ground. When the cord is cut and the insulation is revealed there will be 4 wires: green and white are for data, black is ground and red is Vcc. Connect the black to the ground and the red to the 5V output of the L298N Motor Controller, the white and green can be trimmed so that they are not exposed from the insulation. Heat shrink wraps should be applied around the exposed insulation to small wire section for safety.

Next Modification is to the battery, the deans plug female output must have an adapter that changes it to insulated wires that can be exposed to be attached to the voltage input and ground of the L298N Motor Controller. A male adapter must be fixed to insulated wires, this will be done with the solder. Once that is complete the battery will be able to disconnect and connect from the adapter (NOTE: Make sure to not overuse or apply excessive force to the adaptor or it could be damaged). Finally, strip and tin the wire tips to allow the connection between the adapter and the L298N.

1. **Wiring**

Wiring can be completed using Section V. B. Table 3.

1. **Programming (Initial Structure)**
2. **Port Initializations and Uses**

There will be many different registers and built-in functions used on the Tiva Board to get the proper operation of the autonomous vehicle. The wiring table, Section V. B. Table 3, can be used to see the different port uses and if the pin is input or output, as dictated by the Direction register of the port. PortA will be initialized and used for Pulse Width Modulation (PWM) to drive the motors speed. There will be two separate PWM on PortA, one for the right motor and one for the left motor. Two PWM were used to help keep the bot on a strait path since one motor turns faster than the other with the same PWM. This is due to the efficiency reduction of running a DC motor in reverse. PortB will be initialized and used for Input/Output (I/O) for IR and Ultrasonic sensors. It will also be used to send track direction control using the inverter to output a total of four signals for the L298N H-bridge. PortB will also be configured for interrupts triggered by the front IR proxy sensor (PB2). PortC will be initialized and used for the hall sensors coming from the DC motors. PortD will be initialized and used for the remainder of the Ultrasonic Sensors (NOTE: since most ports have alternate function such as TXD, RXD, Debug, ect. Not all Ports have a full 8 pin out and some components must be split between other Ports). PortF is an optional port but is highly recommended for use of debug and testing. It will be initialized so the SW1 and SW2 are inputs and PF1 – 3 LEDs can be used as outputs.

1. **Other Tiva Built in Functions**

Phase Lock Loop (PLL) and the SysTick timer will be used for accurate delays throughout the program. Both will need to be initialized and called during the start of the main program, just as with any other initialization routines. Internal timer WTimer1A will be used to get time for distance calculations and WTimer0A for pulse counting. PortB will be using interrupts triggered by the front IR sensor (PB2); within this PortB Handler will have the obstacle avoidance routine that will be covered in Section II. B. 4. Trip Distance and Obstacle Avoidance.

1. **User Made Functions for Movement, Turning, and Sensors**

There are many user made functions that are not included with the Tiva Board to achieve proper operation of the autonomous bot. The following functions will be for changing the input to the L298N IN1 – IN4 for direction of the DC motor: Forward, Reverse, Turn\_Right, Turn\_Left. These functions will modify the output of the PB0 and PB1 bits that then travel through the IC inverter to output a total of 4 signals for the IN1 – IN4 pins. Then see Section V. B. Table 4 below for the sequence for the different movement directions. The next move functions will be Right\_Track\_Speed, Left\_Track\_Speed, Move\_Set\_Speed, and Stop functions, these will be altering the two pulse widths in PortA. The Move\_Set\_Speed calls the Right/Left\_Track\_Speed to set the overall speed for the bot. The reason there is a Right/Left\_Track\_Speed function is because there will be one motor will operate slower or faster with the same PWM. This will cause the bot to list to one side or the other. When testing the bot and observation shows listing to the left, the right track is moving faster, and if the bot is listing to the right, the left track is moving faster. To fix this, two PWM will be used by altering the duty of the faster or slower track to get the track to the same speed as the other.

Turning the bot was completed using a time delay that turned the bot ~90°, ± 3°. These functions will use the Move\_Turn\_Left/Right functions shown above to turn their specific direction and the used of a SysTick delay to complete the turn. Testing is important with this application to get the correct time delay of movement to turn the bot 90°.

The final two functions will be to operate the ultrasonic sensors, these functions require the pin used to trigger the specific ultrasonic and the output pin of the same ultrasonic. One function is for the PortA ultrasonics and the other function is for the PortD ultrasonics. Both functions operate the same but use the port specific DATA register to get the I/O and calculate the distance from the Ultrasonics.

1. **Trip Distance and Obstacle Avoidance**

The Move\_Distance function will be made to get a single trip in a straight line from one point to another. This function will use the hall sensors found in the motors to get the proper distance for the trips. Some external required values must be calculated before the function can be created. See Section V. B. Table 5 for the actual calculations of the distance. Make sure to measure the radius or diameter of the drive wheel to the tallest part of the track. Use either the radius or diameter to find the circumference of the drive wheel and track, divide the required distance in the measurement used to find the radius/diameter (For this project, this is done in millimeters) by the calculated circumference. Next multiple the quotient by the number of hall sensor pulses for one rotation of the magnetic sensor on the end of the DC motor. Lastly multiple the product by the gearbox ratio of the DC motor. This can also be acquired by seeing how many times the magnetic disk on the end of the DC motor turns in one motor rotation. Now the Required\_Rotations to complete the trip is found and can be used to drive the bot for the correct distance. Either use a pulse counter with a timer or a software polled counter such as two while loops watching the hall sensor input change from low to high. Once the Rotation\_Count equals the Required\_Rotations the bot should be stopped, this will conclude the Trip Distance function.

The Obstacle Avoidance function is in the PortB Handler, the front IR proxy will trigger this interrupt to stop the bot and start the Obstacle Avoidance routine. The front IR proxy (PB2) is the only trigger for this interrupt handler, and it will immediately stop the bot when the interrupt is triggered. Now using the ultrasonics and other IR proxies the bot will determine what direction to turn to move around the obstacle. When the bot moves, it will record to a global variable how many hall sensor pulses it counted during movement. That way when the next trip is started the bot will subtract or add depending on two flags that are set. One flag is set during the obstacle avoidance routine for if it avoids left (1) or right (0), the other is set before the distance function for the next trip if it turns left (1) or right (0). Once the bot clears the object based on the left/right IR proxy, it will move an addition 279.5 mm to fully clear the object then turn the opposite way it turned during the avoidance to get back on track with the trip. This completes our Obstacle Avoidance function, and is the final function to write into the bot.

Many of the program’s code was altered or changed completely during the testing phase, see section III. D. 3. Testing Changes for the working code changes during the final examination.

**III. DISCUSSION**

1. **Hardware**

Many issues and obstacles were “avoided” during the design of this bot. Initially the power supply was the key problem. Getting a large enough power supply to run the motors, Tiva board, and all our other components was required. Issues with battery housings and mislabeled battery sizes cause some delays to compete our power problem. The initial power source was going to use a batter housing with 3.7V 18650 Li-ion rechargeable batteries but due to a mislabeled battery from the supplier the batteries would not fit into the mount without modifying the mount. This cause the change to the 11.1V LiPo battery used in RC vehicles, this too required modifications and with help from the course instructor the modification was completed. The power supply was attached to the bot and wired through the L298N Motor Controller which had a 5V output that allowed power to be sent to the Tiva board without a smaller voltage source or a voltage divider circuit. The battery is attached, and other sections of the bot could be tested which brought up other issues still caused by our power supply. After creating a test avoidance routine that would drive the bot forward, if the front IR proxy tripped it would turn left and continue till the IR proxy tripped again. Initially the bot would stay inside the interrupt routine with no apparent cause. After using an oscilloscope to see the interrupt signal, micro pulses of ~10µs that didn’t go exactly to zero would still trip the interrupt. The only logical source was interference from the power source so a delay was set to check if the signal was still low when it entered the routine, if so it would run the interrupt, if not immediately leave. The power supply chosen resulted in difficulties for consistent operation. Outputs for pulse-width modulated signals were often distorted or not present until pull-up resistors for these outputs were activated, primarily due to the open-collector method of signal control being the default. The L298N required high and low voltages to be supplied to function properly.

Other minor issues arose from the sensors such as bent sensors, drill damage to IR sensor during mounting, and/or sensors coming free from mounts. The issues caused substantially less delays to the production of the bot than the power issues. One Tiva board had to be re-flashed during the design of the project during the creation of the ultrasonic functions. On the hardware side this caused the bot to have to go through a rewiring that was done incorrectly and had to be fixed since the front and rear components were flipped. Wiring space did become an issue around the female pinout on the Tiva board. Most other wiring was easily maintained under the Lexan platform or under the metal platform keeping the top of the Lexan platform less cluttered for better visual and practical operation. The final issue solved was the improper pulse number and gear ratio for our distance calculations due to an unmarked DC motor that came with the chassis. When using the calculations found in Section V. B. Table 5 the pulses were incorrect and had to be adjusted by taking the percent the bot was supposed to go compared to its actual distance. The final pulses were then multiplied by that percentage to get ~1 m movement for 1 m input to the function.

1. **Programming**

The programming caused far less issues; the one problem was that some Ports had alternate functions thus reducing the number of available pins on that port. Some sensors had to be transferred to different ports to accommodate all the components. This then led to the reflash issue with the Tiva board when creating the ultrasonic function. First attempt was to have the function work for any port and any of the I/O pins, when given the GPIO\_DATA port for either A or D it caused major issues with the Tiva board and even cause one board to need the reflash. This turned the one function into two separate functions for PortA and PortD that essentially did the exact same thing just used their specific GPIO\_DATA. As the project got closer to completion more and more special operations were added such as: a second PWM, more interrupt handlers, and the addition of WTimer0A and WTimer1A. This caused some pins to be moved due to pin specifics needed for these special operations. The other issue that caused some delays was the PWM would not work when in a port that had other components initialized with it. Attempts were made to initialize the port so that all the components would work together but after multiple unsuccessful attempts the PWM was moved to PortA where it could be initialized by itself. Normal programming errors occurred such as syntax and compile errors which were rectified immediately.

1. **Overall**

The overall construction and operation of the bot was a very challenging endeavor. Designing a functional autonomous bot with trips to travel between two points that will do almost every iteration of the course perfectly was daunting. As tests were completed new issues would come up that had to be addressed or fixed if it was hardware. Working with both programming and hardware issues, the hardware issues caused the most delays and overall, the most severe problems. Since programming can be done quickly, loaded, and tested for operation most issues were fixed in less than an hour. If the hardware was damaged or wasn’t initially incompatible more tests would have to be done, technical manuals referenced, and sometimes (most of the time) counseling from the course instructor was needed.

**D. Testing**

The testing process lasted ~24 accumulative hours until the presentation/examination of the operation of the vehicle. During the testing process many changes to the code, hardware, and overall functionality of the platform. In this section we will discuss the overview of the testing process, successes/failures, and the changes implemented.

1. **Overview**

Although the testing process lasted ~24 hours, most of the testing occurred the day prior and day of the evaluation, ~12 hours was dedicated to these times. The testing process delt with a lot of calibration, very small changes to functions and the values inside most functions, and to visual inspection of the vehicle during its test runs. More time should have been dedicated to the testing process, perhaps another 24 hours to have a total of 48 hours accumulated testing time. Trimming down the on the day tests to mainly calibrations and test runs so the vehicle would be ready for the evaluation. During the testing time, changes were recorded but not very efficiently, which is a marked issue to be improved. Better records of proper and improper operation would have helped the testing process. Records such as battery voltage’s correlation to correct distance calculations would have helped show the changes the battery has on our main routines. In the future, spread sheets of data and better record keeping of good and bad operating parameters will be kept.

1. **Successes and Failures**

There were many successes but much more failures during the testing process. The main issue that caused many failed runs of the main routine on the testing course was the distance calculations. It was found that with different voltage charges of our lithium-ion battery would cause the distance calculations to fluctuate. This would cause with too much voltage the distance to be further and with less voltage the distance to be too short for the turns. This could have been caused since the distance function relied on a speed benchmark done before the main routine was initiated. If the voltage was higher the assumption was the motors would have a slightly higher speed and a lower speed with lower voltage. Two things could have possible solved this issue, having a higher voltage lithium-ion battery with a voltage regulator to give us a constant 12V supply that would not change until the battery dipped below 12V. The other change could have been a distance function based off the hall sensor that was talked about in the Section II. B. 4. Trip Distance and Obstacle Avoidance. The issue that arose from this hall sensor function is the improper calculations from the data sheet, which dictated different pulses and possible different gear ratios for the hall sensor. The final found pulses was three high or low pulses per one sensor which could also give six rising/falling signals. This brings up the next issues that caused many failures during the later testing processes.

A lot of the hardware used on the vehicle caused many issues with testing and proper operation. Out of the ten sensors, four ultrasonic and six IR proxy sensors, only two ultrasonics would be used during the examination process. During testing, it was found that the IR proxies would cause interrupts despite being visibly unaffected. The interrupt handler for obstacle avoidance would be activated, waiting a small period of time to check if the interrupt was caused by fluctuating power or by an obstacle. Delays were added to stop the interrupts from triggering but since clocks were used to calculate distances, these delays would cause incorrect clock readings and throw off distance calculations. The IR proxies were completely removed from the code but left on the bot to allow for future modifications, should time have permitted. The ultrasonics were the only functional sensors but that was only if they were in PortA. Initially only one sensor was in PortA and the rest in PortD, but after testing correct operation the ultrasonics would not operate correctly in PortD or PortE. There was enough room in PortA for one more sensor to be coded and this allowed the front and left ultrasonic to work. The final hardware issues were from the battery supply and the motors. The battery supply caused many issues with the vehicle such as improper distance calculations, possible feedback to the ultrasonics, and signal interference to the IR proxy sensors talked about earlier. It is unknown if another power source could have been used without these interferences from the source. The motors cause minor issues to the movement of the vehicle but caused major issues with the distance function and caused the original distance function to change. Out of the four hall sensors only one gave operation signals and one motor was much more efficient that the other.

1. **Changes Implemented**

The initial vision involved several goals: to have the bot record distance covered by each track, to utilize ultrasonics for speed reduction with regards to distant obstacles and directional stability, to utilize infrared sensors for immediate obstacles, and to travel in the desired path. Planning began with the components to be used, and testing of each component began after the construction of the bot was completed. The earliest stages consisted of obstacle detection and linear movement, but problems discovered at these stages persisted through to the final product. It was impossible to create linear movement due to the motors having different efficiencies and the power supply itself changing. This was rectified via several checks in the Better\_Bound function. These fixes were incomplete, however, and introduced new problems to solve when they overcorrected. New solutions require more time to implement and test. Revisiting this project in the future should include: an additional set of speed adjustments in Better\_Bound to account for the overcorrected state, a method to invert the rotational direction of one motor, and checking component functionality at earlier stages. Optional improvements would be to include a PID controller or to use the on-board Quadrature Encoder Interface.

**IV. CONCLUSION**

Design work on any scale is difficult, and problems are sure to arise for any type of design work. Ranging from hardware, software, financial planning, or failed tests that damaged or destroyed the prototypes, many things can go wrong during this process. Knowledge of the material used in the creation of such prototypes and final products is paramount to the success of the platform. Being able to successfully understand the hardware and software behind the platform speeds up the design progress and the final completion of the platform. After the completion of the autonomous vehicle, a sense of understanding could be reached for the design process and for people that have gone through failed design platforms. The course objectives have been revisited, completed, and renewed with the completion of the autonomous vehicle.

**V. APPENDIX**

**A. Screen Captures**

**Schematic, timeline

Description automatically generated**

*Figure 1: Lexan Platform Diagram*

*Diagram

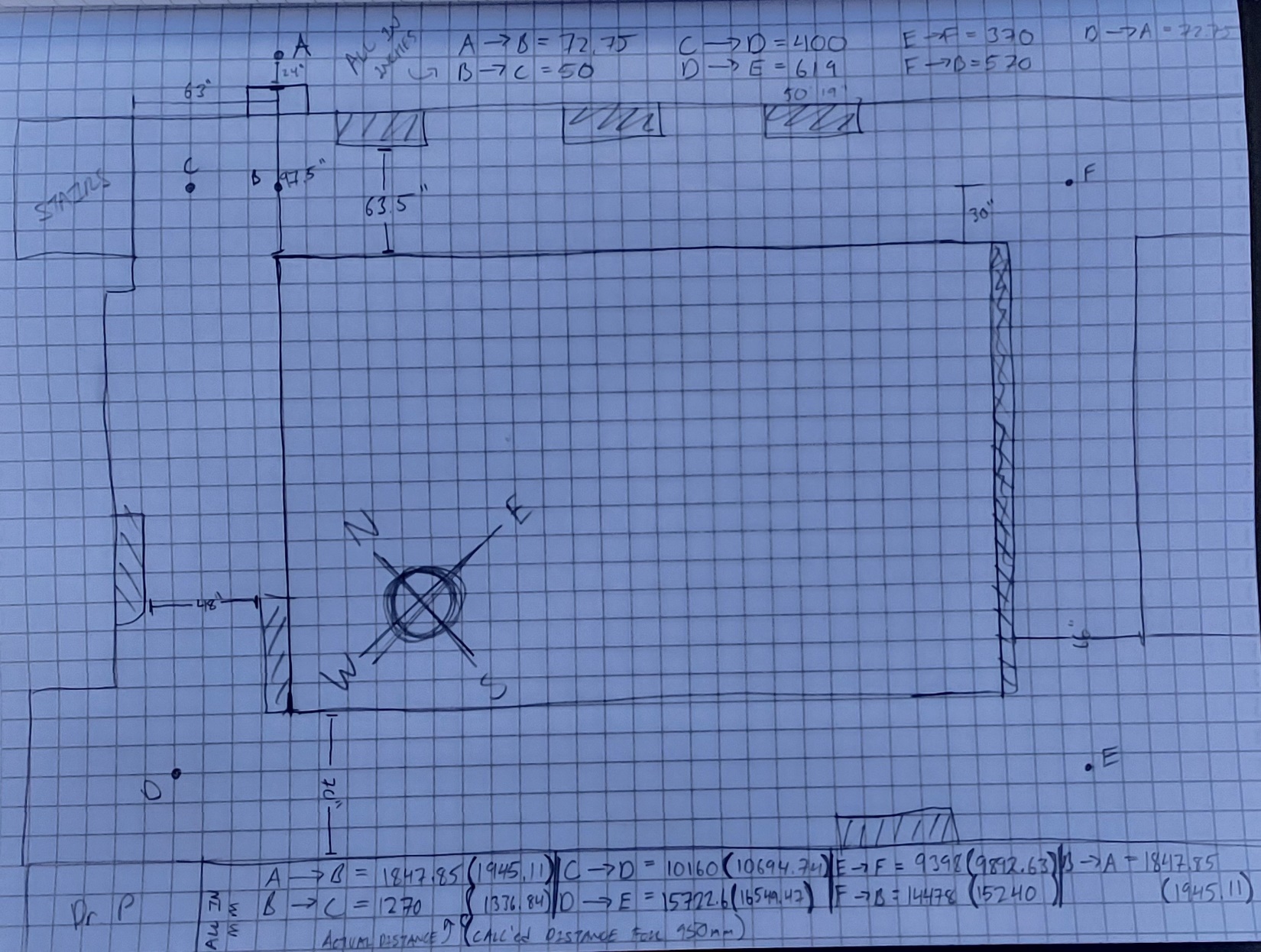
Description automatically generated*

*Figure 2: Metal Platform Diagram*

*Diagram, engineering drawing

Description automatically generated*

*Figure 3: Track Measurement Diagram*

**

*Figure 4: Bot Trip Diagram*

*A picture containing electronics

Description automatically generated*

*Figure 5: Final Bot Construction Image 1*

*A picture containing toy

Description automatically generated*

*Figure 6: Final Bot Construction Image 2*

**B. Tables**

|  |  |  |
| --- | --- | --- |
| **Component Name** | **Component Number** | **Component Value (if any)** |
| Tiva TM4C123GH6PM Microcontroller | 1 | 5V Source Required |
| TS100 Tracked Chassis | 1 | NA |
| DC Motor (Usually Included w/ Motor | 2 | 9V Source Required |
| L298N Motor Driver | 1 | Variable Source (Used 11.1V) |
| Lipo Battery 11.1V (RC vehicle battery) | 1 | 11.1V Output |
| **Sensors, IC’s, and Resistors** | | |
| **Component Name** | **Component Number** | **Component Value (if any)** |
| FC-51 Proximity/Obstacle Sensor | 4 minimum (6 preferred) | 3.3V Source Required |
| HC-SR04 Ultrasonic Sensor | 4 | 5V Source Required |
| 74HC04 IC (Hex Inverter) | 1 | Variable Source (Used 3.3V) |
| Resistor | 3 | 2kΩ |
| 74HC32 IC (2-input OR Gate) | 1 | Variable Source (Used 3.3V) |
| **Mounting and Wiring Components** | | |
| **Component Name** | **Component Number** | **Component Value (if any)** |
| 400-point breadboard w/ 1 ground/power strip | 2 | N/A |
| M3 screws/nuts | - | Lengths(mm): 8, 12, 16 |
| Nylon Round Spacers for M3 screws | - | Lengths(mm): 3 ~ 20 |
| Heat Shrink Tubing | ~3 (personal preference) | Multiple Diameters/Lengths |
| Jumper Cables | - | M to FM : M to M : FM to FM |
| Lexan Sheet | 1 to 3 | 3/16” Thick, Clear, 12” x 12” |

*Table 1: Component List*

|  |  |  |
| --- | --- | --- |
| **Tool Name** | **Tool Model** | **Tool Description** |
| Wire Crimping Tool | WayinTop | Multiple size crimping tool for creating pinned wires Male or Female heads. |
| Wire Stripper | WGGE WG-015 | Used to strip insulation off pre insulated wires, such as jumper cables. |
| Solder Iron Kit | 60W Adj Temp | Heated iron to connect wires or tinning multiple wires |
| Needle Nose Pliers | IRWIN | Pliers with narrow tip for getting into hard-to-reach places |
| Electronic Screwdriver Kit | - | Kit containing multiple screwdriver heads for working on computer screws, usually comes with teasers and plastic punches |

*Table 2: Tool List*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| PORT | I/O (0/1) | USE | NOTES | Other Connections |
| Port A |  |  |  |  |
| PA6 | 1 | PWM1 | RIGHT MOTOR | L298N PWM 1 |
| PA7 | 1 | PWM2 | LEFT MOTOR | L298N PWM 2 |
| PORT | I/O (0/1) | USE | NOTES | Other Connections |
| Port B |  |  |  |  |
| PB0 | 1 | LOGIC1 | RIGHT MOTOR | To 74HC04 Inverter to get two signals: original signal IN1 and inverted signal IN2 on the L298N |
| PB1 | 1 | LOGIC2 | LEFT MOTOR | To 74HC04 Inverter to get two signals: original signal In3 and inverted signal IN4  On the L298N |
| PB2 | 0 | PROX SENS1 | FRONT SENSOR(NORTH) Vcc/Ground goes to 3.3V board | |
| PB3 | 0 | PROX SENS2 | RIGHT SENSOR(EAST) | Vcc/Ground goes to 3.3V board |
| PB4 | 0 | PROX SENS3 | REAR SENSOR(SOUTH) | Vcc/Ground goes to 3.3V board |
| PB5 | 0 | PROX SENS4 | LEFT SENSOR(WEST) | Vcc/Ground goes to 3.3V board |
| PB6 | 1 | ULTRASONIC OUTPUT3 | REAR ULTRA(SOUTH) | Vcc/Ground goes to 5V board |
| PB7 | 0 | ULTRASONIC INPUT3 | REAR ULTRA(SOUTH) | - |
| PORT | I/O (0/1) | USE | NOTES | Other Connections |
| Port C |  |  |  |  |
| PC4 | 0 | HALLSENSEA1 | Front Motor | Goes to the 5V board and ran through the voltage divider circuit to lower the signal voltage to under 3.3V |
| PC5 | 0 | HALLSENSEA2 | Front Motor | Goes to the 5V board and ran through the voltage divider circuit to lower the signal voltage to under 3.3V |
| PC6 | 0 | HALLSENSEB1 | Rear Motor | Goes to the 5V board and ran through the voltage divider circuit to lower the signal voltage to under 3.3V |
| PC7 | 0 | HALLSENSEB2 | Rear Motor | Goes to the 5V board and ran through the voltage divider circuit to lower the signal voltage to under 3.3V |
| PORT | I/O (0/1) | USE | NOTES | Other Connections |
| Port D |  |  |  |  |
| PD0 | 1 | ULTRASONIC OUTPUT1 | FRONT ULTRA(NORTH) | Vcc/Ground goes to 5V board |
| PD1 | 0 | ULTRASONIC INPUT1 | FRONT ULTRA(NORTH) | - |
| PD2 | 1 | ULTRASONIC OUTPUT2 | RIGHT ULTRA(EAST) | Vcc/Ground goes to 5V board |
| PD3 | 0 | ULTRASONIC INPUT2 | RIGHT ULTRA(EAST) | - |
| PD6 | 1 | ULTRASONIC OUTPUT4 | LEFT ULTRA(WEST) | Vcc/Ground goes to 5V board |
| PD7 | 0 | ULTRASONIC INPUT4 | LEFT ULTRA(WEST) | - |
| PORT | I/O (0/1) | USE | NOTES | Other Connections |
| Port F |  |  |  |  |
| PF0 | 0 | SW2 |  |  |
| PF1 | 1 | RED LED |  |  |
| PF2 | 1 | BLUE LED |  |  |
| PB3 | 1 | GREEN LED |  |  |
| PF4 | 0 | SW1 |  |  |

*Table 3: Wiring Table*

|  |  |  |  |
| --- | --- | --- | --- |
| **PB0 Output** | **Inversion** | **Bit Code(PB0, ~PB0, PB1, ~PB1)** | **Bot Movement** |
| 0 | 1 | 0101 | Reverse |
| 1 | 0 | 1001 | Left (In place turn) |
| **PB1 Output** | **Inversion** | 0110 | Right (In place turn) |
| 0 | 1 | 1010 | Forward |
| 1 | 0 |

*Table 4: Track Movement Table*

|  |  |  |
| --- | --- | --- |
| **Variables** | **Variable Values** | **Calculation** |
| D = Diameter | D = 47.88 mm (measured) |  |
| C = Circumference | C = 150.419 mm (calculated) |  |
| i = Distance | i = 1000 mm (1 m, given, user) |
| x = pulses for hall rotation | x = 12 (given, datasheet) |
| y = gear ratio | y = 75 (given, datasheet) |
| z = final pulse count for distance | z = 5983.29 hall sensor pulses (calculated, for 1000mm) |

*Table 5: Distance Calculations (Initial Distance Function, not in final code)*