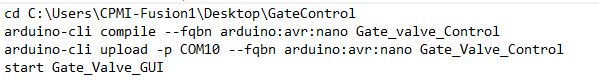
Gate Valve Control

# How To Use

The valve control software is built to be easy to use. A single executable batch file under the name “Gate Control” with an icon of a valve exists on the desktop of the computer. When the arduino is connected to the computer, and the power source to the wall, note that it must be connected to COM10, as this is where the program is set to look for the arduino (more notes on this later), simply run the executable. Plugging the power source after the running of the program will not cause issues, however it is possible that if you try to use the software to open valves before power is supplied you may flip the state of the buttons. If this does happen, simply restart the program and it will return to normal.

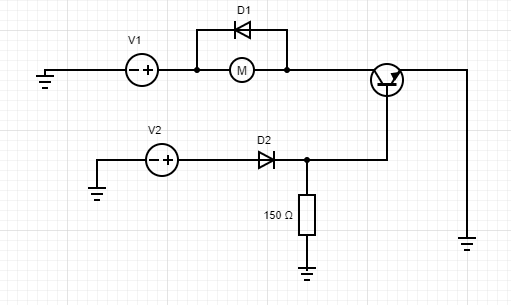
# The Executable



The executable file relies on the “arduino-cli”. This is a command line extension that allows for arduino.ino files to be compiled and uploaded to your board through the command line. The image above is the entirety of the code for the executable. It enters the file the code is in, compiles it with the specifications of our board, uploads it through COM10, and runs the executable for our interface. One of the reasons use of COM10 is required is due to this script explicitly accessing it, as does the interface when attempting to communicate with the arduino.

# 

# Circuit Schematic & Operation



* This is the circuit that controls each motor individually, the circuit that controls the system of motors is merely this duplicated for each motor being controlled
* V1 represents the DC 24V power source the circuit runs on and V2 is the DC voltage the arduino sets in order to control the circuit.
* D1 and D2 are both equal value diodes, D2 is to ensure that no feedback goes to the arduino and possibly frying its circuitry. D1 is required to to the gate valve (represented by the M for motor) being a solenoid which when switched off produces high current feedback. This feedback is then fed into the diode instead of allowing it to potentially fry the transistor. Use of a diode with a low threshold voltage is required, as the threshold voltage of the transistor is around 4V. Should the voltage drop of D2 be too high the arduino would not be able to operate the transistor as a switch.
* How the circuit wholistically works is the valve is always powered up to 24V, however it isn't grounded. The portion connected to the arduino sets the base of the transistor to ground, and as such current cannot flow from collector to emitter, I.E, no current can flow through the motor. When the arduino supplies 5V, instead of its steady state 0V, the base of the transistor is no longer grounded, and current is allowed to flow through the valve effectively putting it in its “ON” state. When the arduino is switched back to 0V output, the valve returns to its “OFF” state.

# Arduino Program Control

The code uploaded to the arduino performs a couple of tasks. It initially sets the pins used in the circuit to “output” mode, and sets all their outputs to zero to ensure we start in the “OFF” state for all valves. It then initializes “serial communication”, meaning it starts the process in which it allows communication with the GUI on the computer. It then continuously checks what value the GUI sends to the arduino, and switches the associated pin “ON” (5V) or “OFF” (0V) depending on which state the pin previously in. If the pin was off, when the associated value is read it will turn on, and vice versa. Further comments and explanation is available in the source code file itself.



This is the function that initializes and allows for communication between the arduino and the GUI set up on the computer. The value “9600” is the bandwidth said communication takes place on. It is a relatively arbitrary value as communication can occur on any bandwidth, however it is important to note that “9600” is the standard for serial communication, and that the GUI is set to communicate on the same bandwidth. Hence if communication on a different bandwidth becomes favorable it must be changed for both the code running on the arduino and the source code of the graphical interface.

# Valve Control Interface

The interface is in charge of allowing the computer to communicate with the arduino in a manner that is self explanatory to any and all of its users. It operates through initializing communication with the arduino through a specified port and on a specified serial bandwidth. The source code of the GUI mainly consists of “drawing” toggle buttons, buttons that hold an “ON” and “OFF” state that can be seen through a color variation of said buttons. These buttons are later programmed to send specific characters to the arduino depending on the button pressed. The arduino receives these character inputs and “turns on” the respective motor. It is good to note that the characters parsed by the GUI are arbitrarily chosen for the sake of ease of programming, however these values must align with the code on the arduino as so that the system can function as expected.



The second line is what initializes communication with the arduino. If the bandwidth on which we wish the two to communicate should ever change from the standard “9600”, it must be set both here and in the code the arduino runs. Moreover, if we ever wish to change the port the arduino is on it is very important to change the “COM10” to the respective COM the arduino is connected to. This is because the interface does not naturally recognize the arduino as connected, but searches for it on the specified port. Additional information is also available on the source code under the name Gate\_Valve\_GUI.