Project Overview

Written by:

Socorro Alejandra Montelongo Loredo

Arturo Urias Jiménez

Gabriel Enrique Lascurain Flores

Ana Carolina Coronel

Yumee Chung

Victoria Lilian Robles Vargas

Josemaría De Alba Arguelles

Samantha Erin Medina Muñoz

Ricardo Ruiz Cano

Índice

Hardware Design

In this design, three relays were used to simulate the behavior of a sensor. The relays allow switching between different input signals (such as reference voltage, ground, or disconnection) in a controlled manner, replicating real conditions that a sensor could generate during operation.

The objective of the design is to implement a signal selection system using relays, controlled by an ATmega328P microcontroller. This system allows selecting between different inputs and directing a single output signal, ensuring the integrity and strength of the original signal.

The circuit consists of three relays that work as electrically controlled switches. The first two relays act as signal selectors. Each has two inputs: one connects to a sensor and an open (disconnected) line, while the other connects to 12 V and ground from the power supply.

The outputs of these relays are directed to a third relay, which acts as the final selector. This last one changes state through a 5 V signal sent by the microcontroller, allowing dynamic selection of the output signal depending on system conditions.

Relays were chosen instead of a conventional multiplexer because they allow direct connection between signals, ensuring better current conduction. Unlike multiplexers, which replicate the signal with very low current, relays maintain the necessary intensity to ensure that the signal is properly detected by the receiving system.

The following are the main electronic components used in the circuit design:

2N2222 transistors: These function as electronic switches. The microcontroller cannot supply enough current to directly activate the relay coil, so it sends a small signal to the transistor base. This allows the transistor to conduct current and activate the relay.

1 kΩ resistors: These are placed in series with the transistor base to limit the current flowing to that terminal. This protects both the transistor and the microcontroller from excess current.

1N4148 diodes: These act as flyback diodes, conducting the induced current when the relay turns off and preventing voltage spikes from damaging the transistors.

THD-1201L relays: These allow controlling higher power or different voltage circuits by switching the signal path via coil activation. They are driven by transistors that receive control signals from the microcontroller.

ATmega328P microcontroller: Responsible for generating the digital control signal that determines the state of the main relay and selects the desired output signal.

12 V and 5 V power supply: Provides the energy required for relay activation (12 V) and microcontroller operation (5 V).

The system uses three relays to simulate sensor behavior and select different input signals in a controlled way. Each relay acts as an electrical switch that connects or disconnects specific signals based on commands sent from the ATmega328P microcontroller.

The first two relays serve as primary selectors:

The first relay switches between a real sensor signal and a disconnected input (open circuit).

The second relay alternates between a 12 V signal and ground from the power supply.

The outputs of these two relays are combined and directed to a third relay, which acts as the final selector. This third relay is controlled directly with a 5 V signal from the microcontroller and determines which of the combined signals will be sent as the system's output.

Thanks to this configuration, the microcontroller can dynamically select which signal to simulate or transmit, replicating different conditions that a sensor might generate during real operation. Using relays guarantees that the signal maintains the integrity and strength necessary to be correctly detected by other devices connected to the system.

T1	**T2**	**T3**	**Salida**
0	**0**	**X**	**+12 volts**
0	**1**	**X**	**Gnd**
1	**X**	**0**	**Sensor**
1	**X**	**1**	**Circuito abierto**

The table shows the possible combinations of control signals sent to the three relays (T1, T2, and T3) and the corresponding system output signal.

When **T1 = 0** and **T2 = 0**, regardless of the state of **T3 (X)**, the output is a fixed **+12 volts** signal.

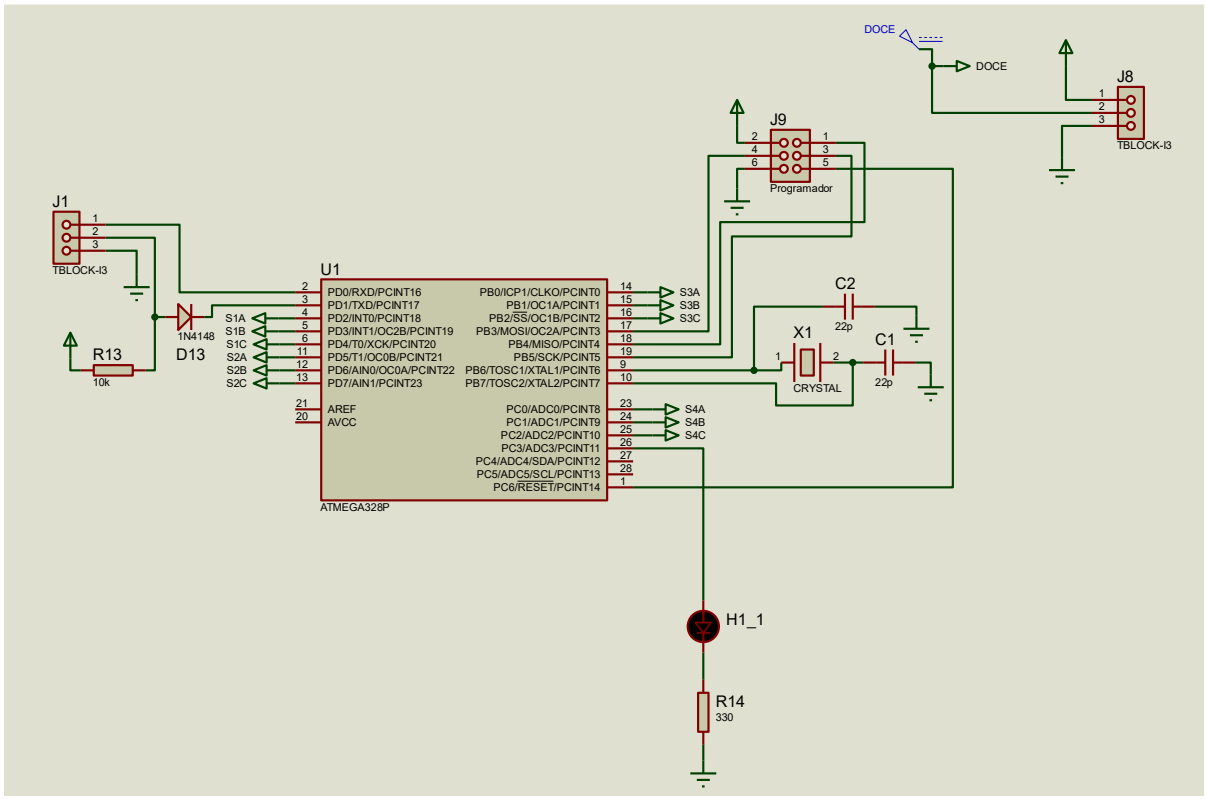
If **T1 = 0** and **T2 = 1**, regardless of the state of **T3 (X)**, the output is connected to **ground (GND)**.

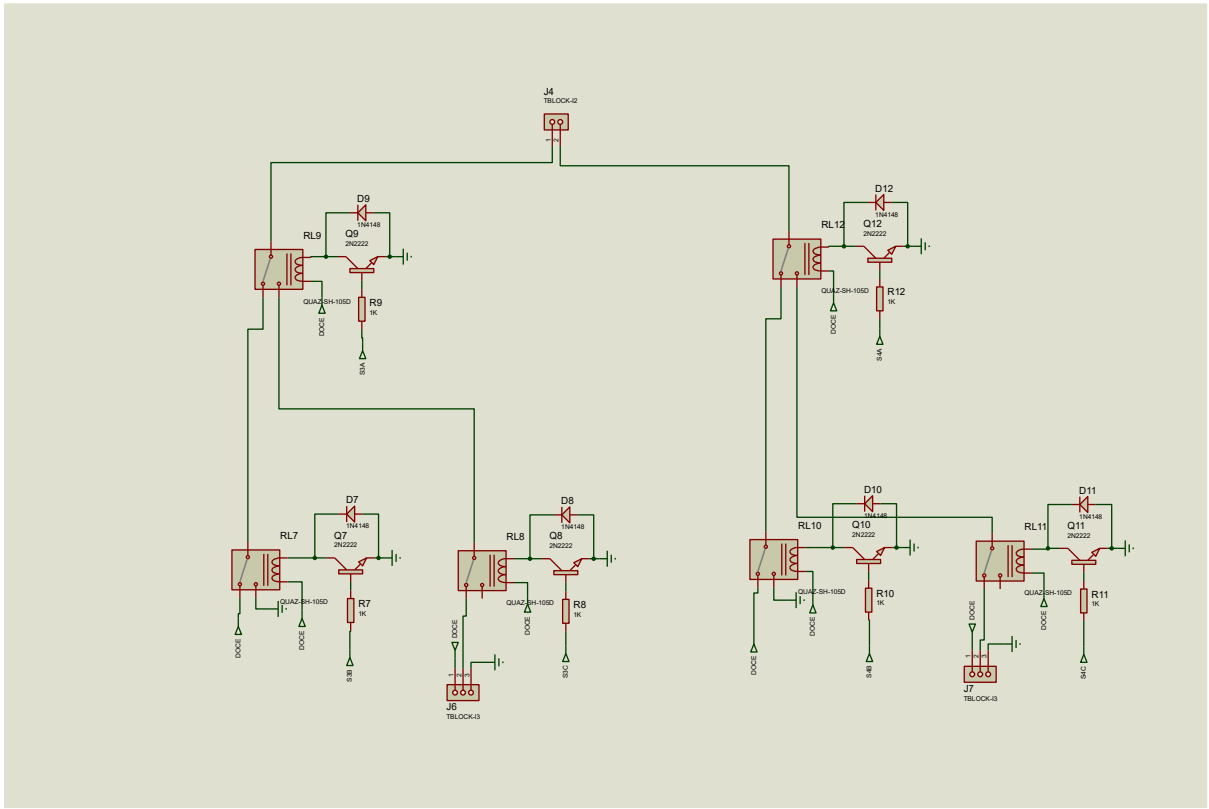
When **T1 = 1** and **T3 = 0**, regardless of the value of **T2 (X)**, the output corresponds to the **sensor signal**.

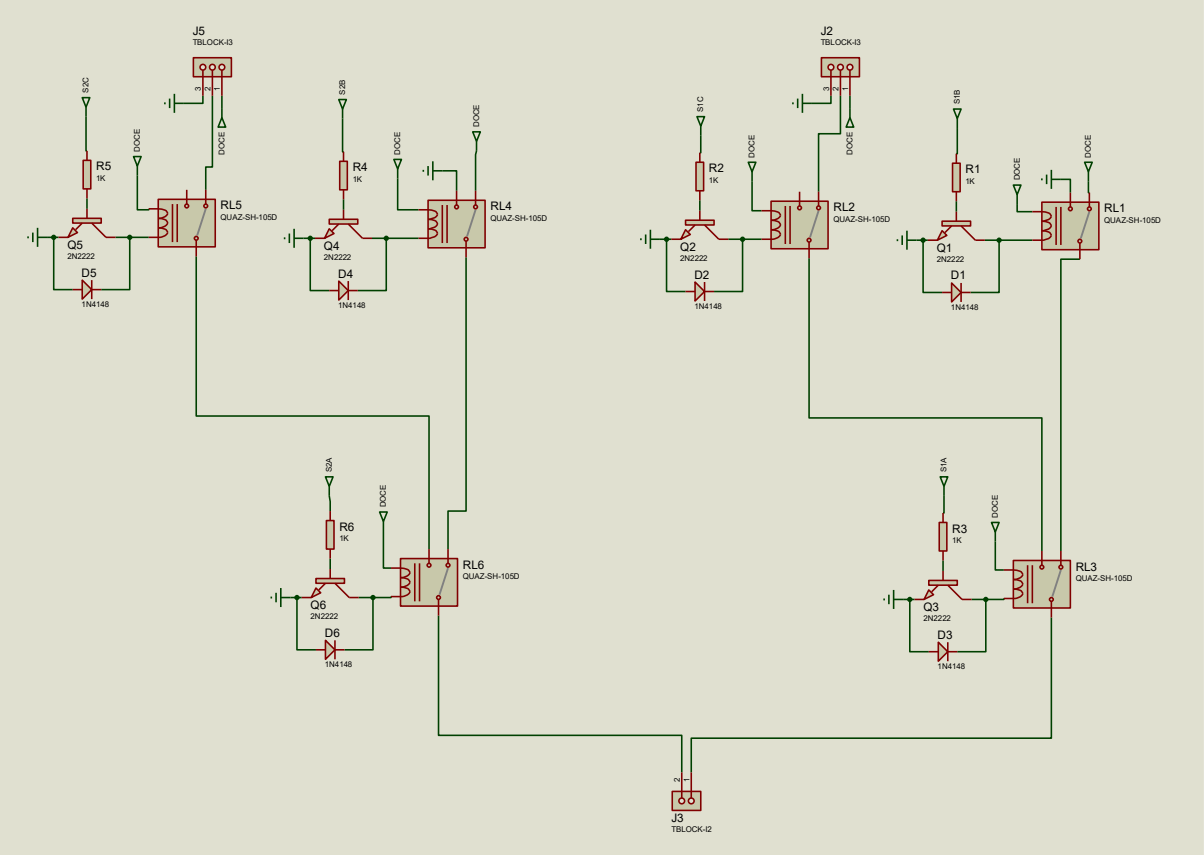
Finally, if **T1 = 1** and **T3 = 1**, regardless of the state of **T2 (X)**, the output is an **open circuit**, simulating a disconnection or lack of signal.

This table summarizes how the control combinations determine the signal that is simulated or sent to the output, allowing different operating conditions to be replicated through relay control.

Hardware Schematics







Hardware Setup

For the assembly of the prototype, a breadboard was used as the mounting base, allowing components to be easily connected and rearranged during testing. Below is a step-by-step description of the assembly process and the connections made:

Step-by-step assembly:

Relay installation:
Three relays were placed on the breadboard. These are responsible for simulating the behavior of a sensor by selecting between different input signals (12 V, GND, sensor signal, or disconnection).

Connection of 2N2222 transistors:
Each relay was controlled by a transistor configured as an electronic switch. The base of each transistor was connected through a 1 kΩ resistor to the corresponding microcontroller pin.

Insertion of 1N4148 diodes:
A diode was placed in reverse-parallel with each relay coil to protect the transistor from voltage spikes generated when turning off the relays.

Microcontroller connections:
The ATmega328P microcontroller was programmed to send control signals to the three transistors, determining the state of each relay based on simulated conditions.

Input signal connections:

- A real sensor signal and an open (disconnected) line were connected to the first relay.
- The second relay was connected to 12 V and GND from the power supply.
- The outputs of these two relays were directed to the third relay, which acts as the final signal selector.

Power supply:

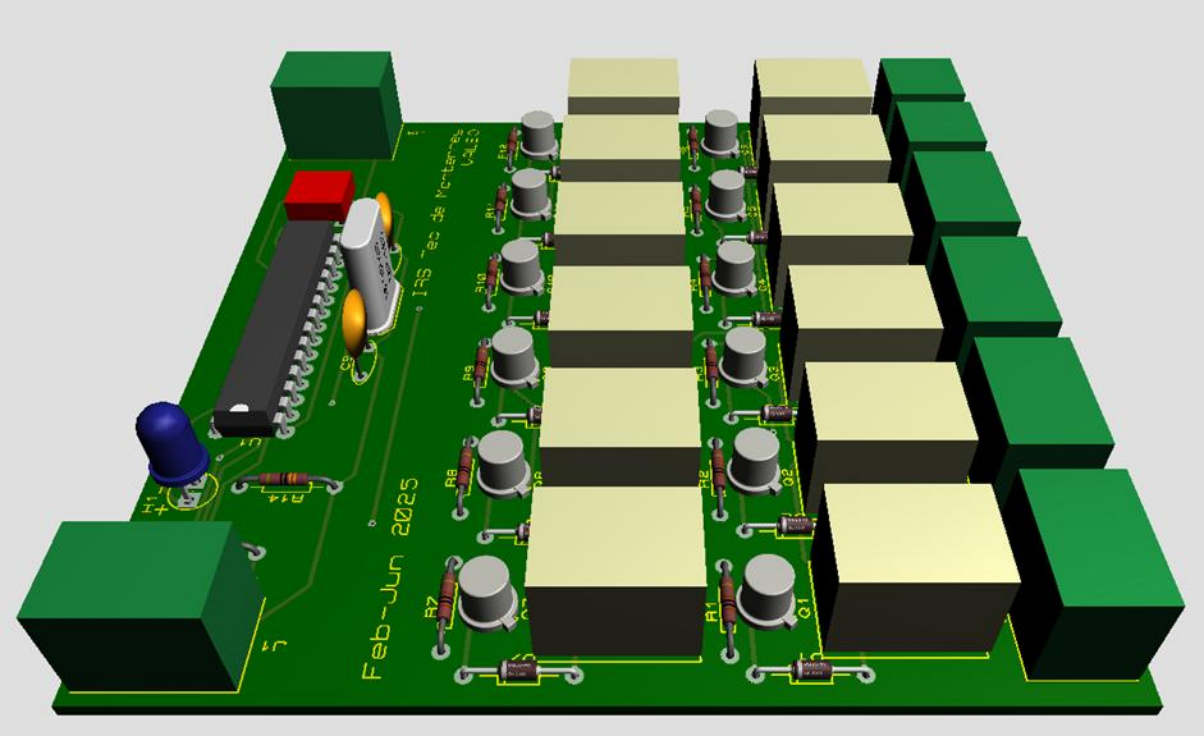
The circuit was powered by an external 12 V DC and 5 V DC source for the main signals and relays.

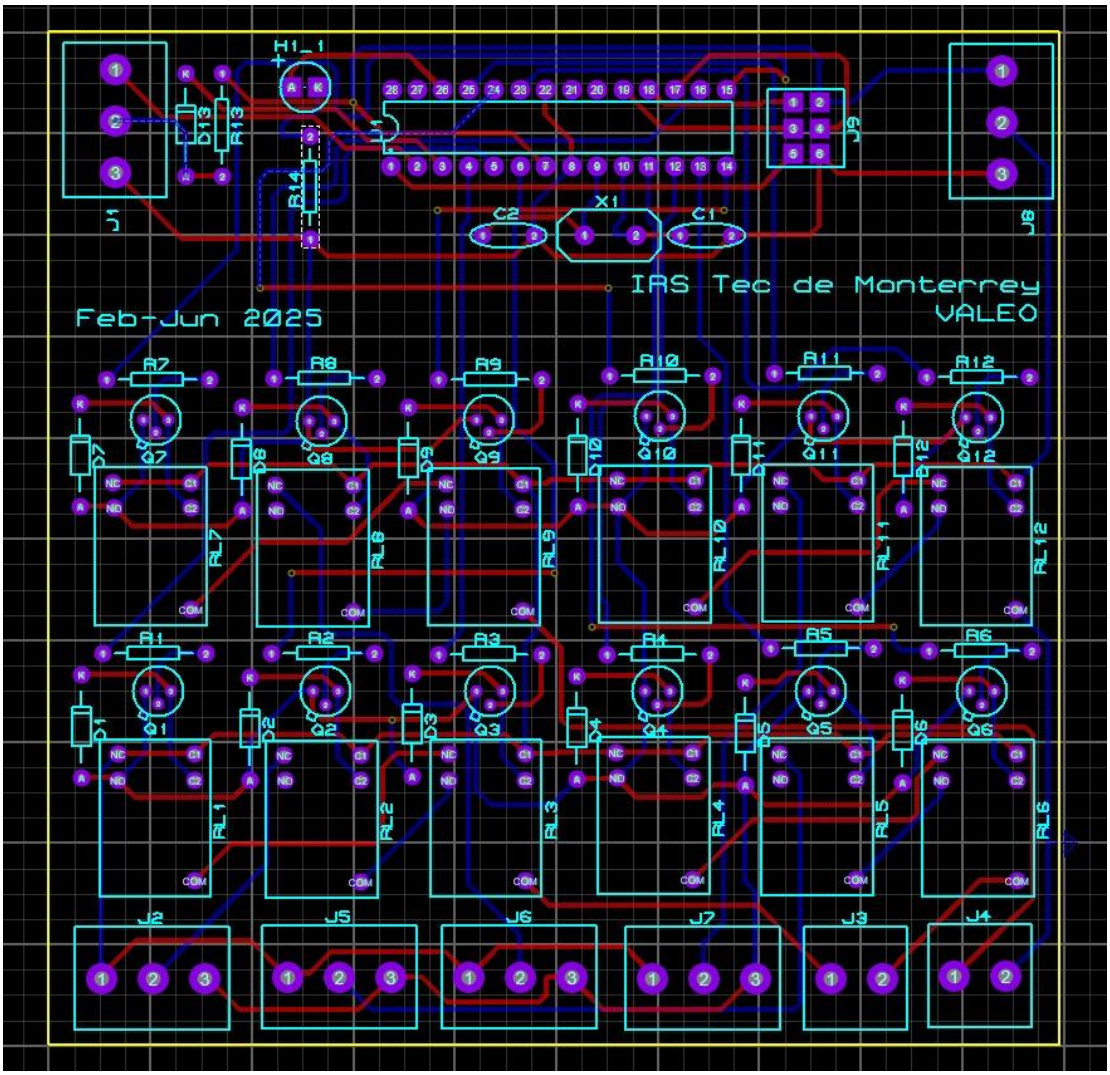
Adjustments made:

During the assembly, it was necessary to rearrange some connections on the breadboard to ensure proper isolation between input signals. Relay activation times were also adjusted in the microcontroller code to ensure precise switching and avoid bouncing or signal errors.

Photographs of the final breadboard assembly are included, showing the location of the relays, transistors, resistors, and diodes, as well as the connections to the microcontroller.
Insert photo

Below is the PCB design we created for the circuit. This design has already been sent for manufacturing; however, due to time constraints, we do not yet have the physical PCB for implementation. Therefore, all testing and validation were carried out only on the breadboard prototype.


IAS Tec de Monterrey
VALEO
Feb-Jun 2025


IAS Tec de Monterrey
VALEO
Feb-Jun 2025

Software Design

Software Architecture

Our project uses a layered architecture pattern where our components are separated in layers of subtasks:

We have a unique layer for microcontrollers which is programmed to modify the signals sent to the different relay sets in order to be able to use a terminal in order to modify what signals are turned on and off for every sensor. This allows us to remotely control which sensors we want to activate and what state we want to test in order to simulate status without having to modify the hardware. This is used in order to reduce risk of accidentally disconnecting or damaging the components that would be moved in order to change the status.

Then, we have the presentation layer which is in charge of the interface for the end-users. The users of the application can interact with four different buttons presented on their view: V for short to voltage, G for short to ground, C for open circuit, and finally, S for connected which refers to the normal state of the sensors.

Software Development Dependencies

Microcontroller:

Interface/Serial Port:

- Python
    - Python Serial Port Extension
    - Uvicorn
    - FastAPI

Software Code Documentation

Microcontroller:
Link

Interface/Serial Port:
Link

Software Usage

Microcontroller:

The code utilized by the user is very simple to use. It uses a combination of a number and a letter in order to send different combinations of on and off commands in order to simulate the signals sent to the relays to control their status. First you send the number of the

sensor you want to modify, then next to it you send one of four letters in order to select the state you want to simulate.

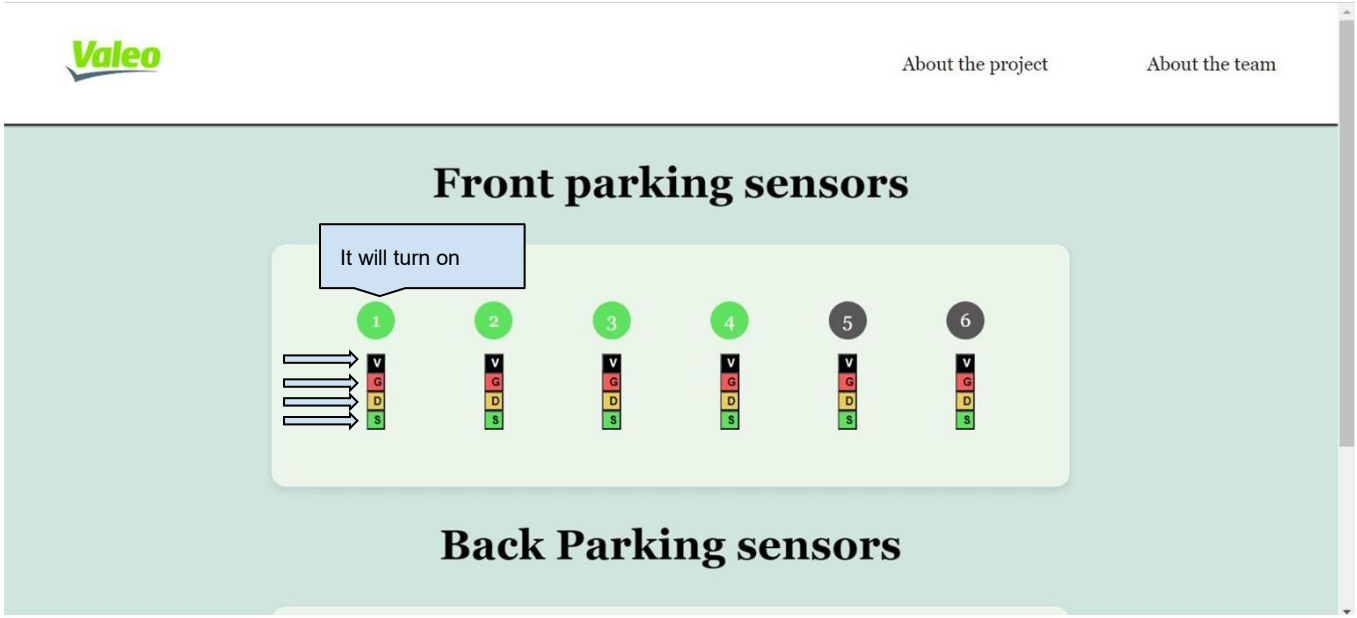
Letter	State
V	Shortage to Battery
G	Shortage to Ground
S	Sensor working normally
C	Open Circuit

You can send any of these characters after a number in order to change the status of the sensor. The character can be sent in both uppercase and lowercase and still work the same way for any sensor. If you send an invalid combination or invalid character, you receive a message warning you of an invalid combination, this does not modify any sensor value. You can send as many characters and numbers as you want but the terminal will only accept a valid combination of letters. This is done by only reading the final 2 characters which must be a valid number and character in order to do something.

Command Example	Result
1V	Set sensor 1 to short to battery
2G	Set sensor 2 to short to ground
3S	Set sensor 3 to work normally
4C	Set sensor 4 to an open circuit

Interface/Serial Port:

For the usage of the interface, we have to run the main code. The app.py file which receives commands from the HTML interface and forwards them through the serial port to the microcontroller. In other words, it manages communication between the user and the hardware (via COM). To run this code in the terminal: python '*app.py*'



Depending on the sensor connected the *green light* will turn on. If the *V* button is pressed, it will send short to voltage, if the *G* button is pressed, it will send short to ground, if the *D/C* button is pressed it will send open circuit, and finally if the *S* button is pressed, it will send the normal state.