ME 460/461: Documentation

Team 21: Vehicular Laboratory Automation Device (VLAD)
Fall 2022 - Spring 2023
Boston University

Adam Bahlous-Boldi
Kyle Fieleke
Miguel Ianus-Valdivia
Vlad Pyltsov
Abin George

Overview

The VLAD (Vehicular Laboratory Autonomous Device) is a system that aims to automate laboratory workflows, eliminating the need for constant human intervention and saving valuable time. The goal of this report is to provide a detailed description of the design, assembly, and testing of the VLAD system.

The project was the brainchild of Professor Keith Brown, who needed an efficient and flexible laboratory automation system for his own Laboratory. Professor Brown runs the KABlab, which designs and tests hierarchically structured soft materials. Part of their research is about developing testing systems to be able to systematically and autonomously analyze various material properties of developed structures and materials. However, current testing systems lack modularity and would be challenging to adapt to different laboratory spaces or different experiments. The VLAD aims to help tackle this, by allowing for spacial flexibility in setting up autonomous experimental systems.

This document aims to provide a detailed description of the design, assembly, and testing of the software aspect of the project.

More information and the final report of this project can be found here:
`https://github.com/abingeorge07/ME461-Vehicular-Lab-Automation-Device`

IDE Installation

For this project, we used the Espressif IDE that allows to program the ESP32 microcontroller. The IDE can be downloaded from the following link:
`https://docs.espressif.com/projects/esp-idf/en/latest/esp32/get-started/index.html#installation-step-by-step`

Building the Project

Once installed, open the powershell application named *ESP-IDF 5.1 PowerShell*

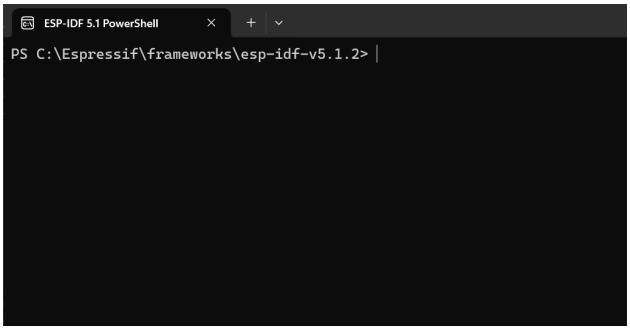

Figure 1: ESP-IDF 5.1 PowerShell

Run the following command to setup the environment for the *ESP32* microcontroller.

```
idf.py set-target esp32
```

Then, navigate to the project folder and run the following command to build the project.

```
cd <project_folder>
idf.py menuconfig
```

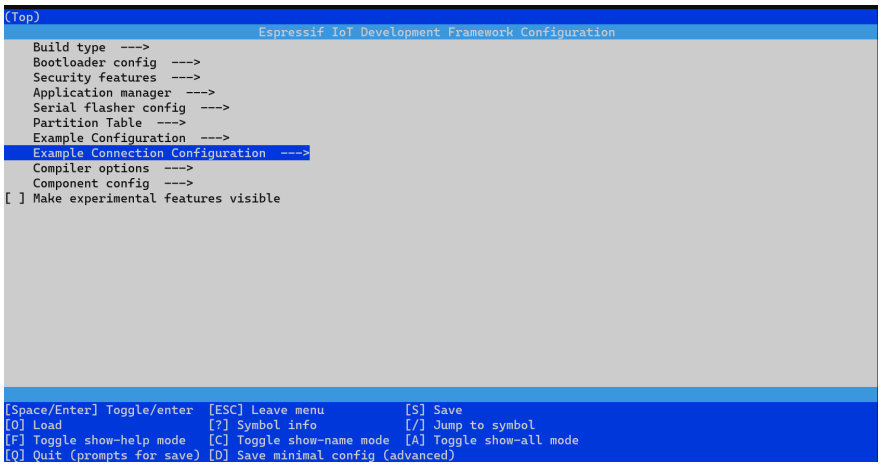

Figure 2: Menuconfig

Under **Example Connection Configuration**, you can set the SSID and password of the netwrork you want to connect to.

Under **Example Configuration**, you can set the IP address and port number of the UDP server.

Once this is done, run the following command to build the project. This may take a few minutes if it is the first time building the project.

```
idf.py build
```

Flashing the Project

Once the project is built, run the following command to flash the project to the microcontroller. The port is the port number of the computer to which the microcontroller is connected to. This can be found in the device manager for Windows. Conversely, if not specified, the IDE will automatically detect the port.

```
idf.py -p <port> flash
```

NOTE: Depending on which ESP32 version you have, you may have to hold down on the reset button while flashing the project.

Monitor Output

If testing and you want to see the output of the microcontroller, run the following command.

```
idf.py -p <port> monitor
```

UDP Communication Parameters

Following are the network setting set on the ESP32 microcontroller.

SSID: kablab
Password: factorio

Following are the UDP communication parameters.

Port: 3333

IP Address is currently dynamic, meaning every time the ESP32 is connected to the network, it will be assigned a different IP address. Currently, we are working to make the IP address static.

Current Network Setup

Currently, we create that network using a laptop by enabling the hotspot feature and setting the SSID and password to the ones mentioned above. Once the ESP32 is connected to the network, the ESP32's IP address can be found under the *Hotspot* section of the network settings.

Issues

If you face any bugs, please create an issue on the GitHub repository.