Enabling Fast Charging with Energy Storage

Dr. Jonathan Kimball, Dr. Alvaro Cardoza, Kartikeya Veeramraju, Evonne Siampos

Department of Electrical and Computer Engineering

Funded by Bitrode

Introduction:

- With EVs on the rise, fast-charging capabilites are more important than ever. The only problem is the inconvenience of charging them in comparison to gas-powered vehicles. Currently, it takes fast charging EV stations anywhere from 20 min to an hour to charge to 80% capacity.
- Our research works to give all the essential power to an EV within the time it takes to fill up a tank of gas (10 min). With this, there come obstacles:
    - Heat generated by such a fast charge
    - Amount of electricity pulled from the grid
    - Ability to connect to the medium-voltage network

Approach:

- Tools used: EV batteries, ESS (Energy Storage System), Raspberry Pi, PowerPC, C2000 micro-processors, 500kW Power & Voltage Packs
    - ESS: temporarily stores energy and release it gradually to avoid power
    - Raspberry Pi: receives data from EV batteries, ESS, and micro-processors to determine the power that should be supplied from the charging station - this infromation is sent to the PowerPC
        - Value of reactive current is also sent to the micro-processors to offer grid support
    - PowerPC: controls the Power & Voltage Packs based on the data received from the Raspberry Pi
    - C2000 micro-processors: manages communication and power flow between the charging station and the EV based on the reactive current, which ultimately offers grid support

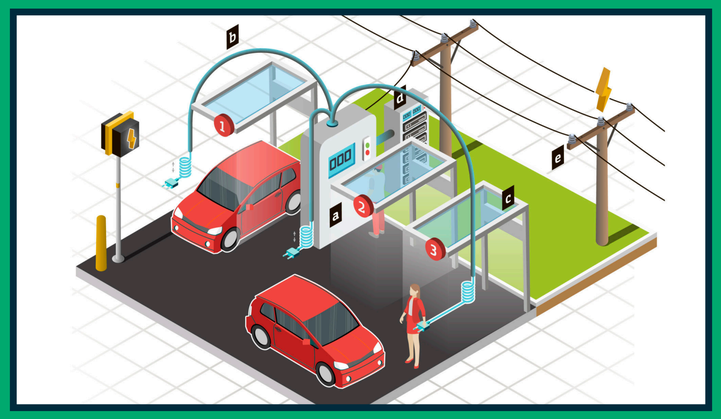

Figure 1: Ideal, real-world, Implemtation

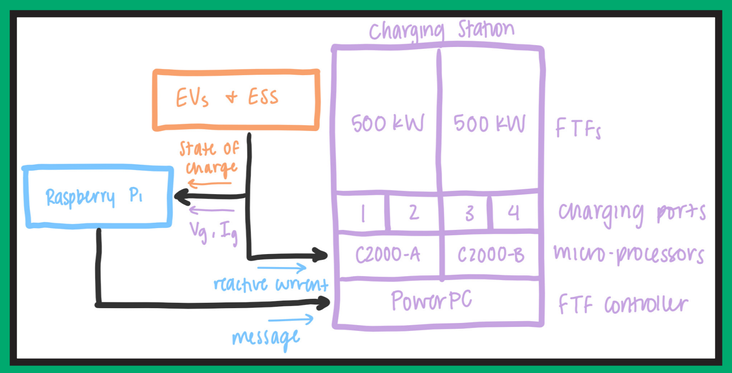

Figure 2: Hardware Overview

Conclusion:

Unfortunately, the whole project is not finished. Thus, there is not much to say about the overall outcome of the research project. However, there is something to say about the results of the Raspberry Pi code that I developed, which is finalized. To summarize, given different pieces of data that each required distinct messages to be assembled, the program correctly assembled the needed message and accurately communicated it with the PowerPC. This means that the power supply was given the right commands to adjust the voltage and current to maintain the most efficient and safe charging experience.

Personal Contribution: Raspberry Pi Code

- Objective: to send data to a PowerPC based on information that was received from EV, ESS, and micro-processors
- Functions:
    - UDPinit: initializes the socket (communication channel) to send/receive data
    - RBP_constructor: takes the given values and organizes them into an effective structure; with these values, the code calls one of the following functions:
        - RBPSendSetPt: assembles message to set or get data, including current voltage, power , and current
        - FTFToggle: assembles message to run or stop charging
        - RBPSetRemote: assembles message to set or clear the remote state
    - SendUDPMsg: called after assembling the message, this sends the message over the open server to the PowerPC

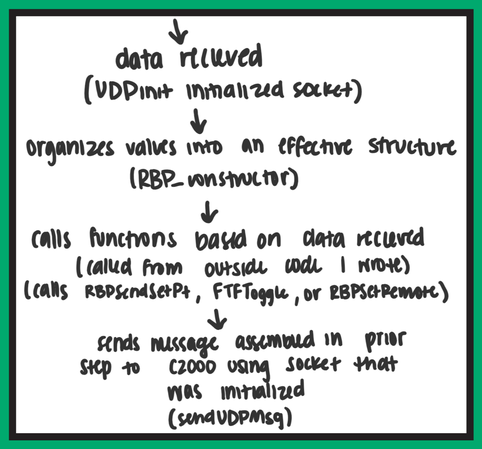

Figure 3: Raspberry Pi Code Overview