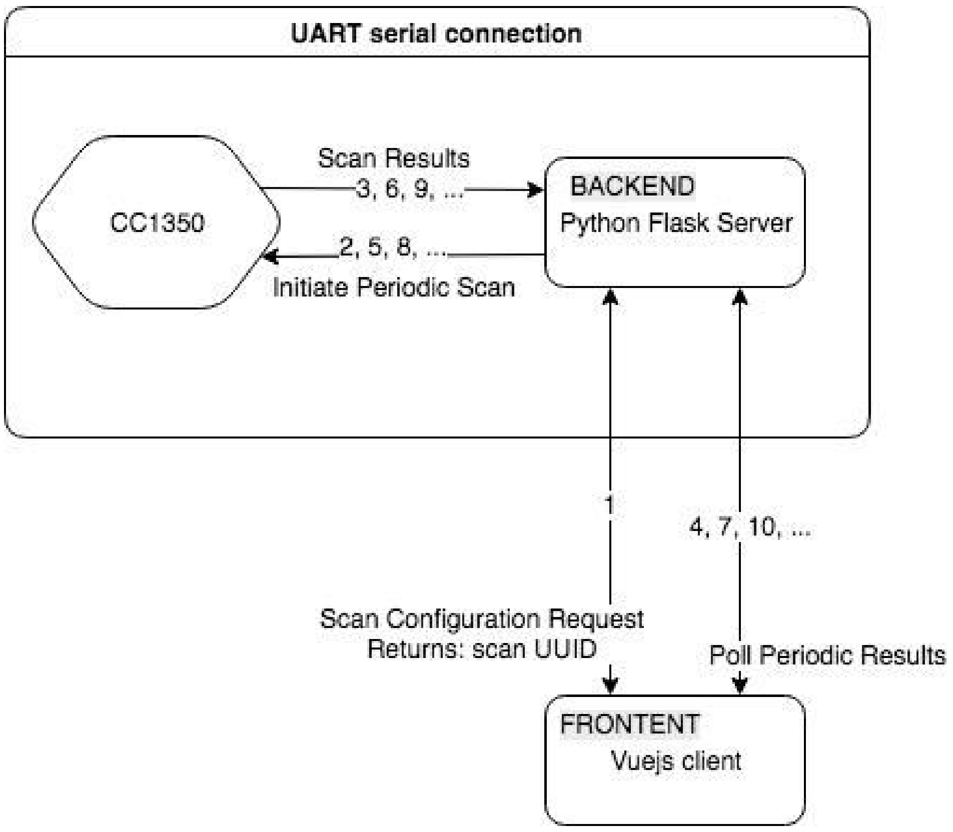
Apr 2018
Liad Wainberg 201324332
Assaf Cohen Shahar 304866163

IOT project - RF scanning and analyzing

Our project was to implement a Spectrum analyzing tool using a TI CC1350 board. Using our analyzing system, one can scan different frequency bands in different configurations in order to spot quiet frequencies and times in the day for future use in other projects.

High Level Design

The system's UI is a simple web page in which the scan parameters can be configured and the scan results are displayed (both in raw and aggregated forms). The system's server manages different scans and connects VIA a serial connection to the CC1350 board which performs the scans themselves.



Main utilities

1. **CC1350:**
   - Scans frequencies upon request.
   - Receives scan parameters:
     - Bottom frequency (Lowest frequency in the range to scan)
     - Top frequency (Highest frequency in the range to scan)
     - Step size (Scan resolution)
     - Scan loops (Amount of times to sample each frequency)
   - Returns scan results after every single sample, specifying frequency and RSSI value.
   - All communication is UART based.
   - Supports scanning frequencies in the ranges 431Mhz - 527Mhz and 861Mhz - 1054Mhz (Sub-1Ghz bands supported according to TI specs).
2. **Backend:**
   - Python Flask Server.
     - Need to install python packages:
       - Flask
       - Pyserial
       - Numpy
   - Aggregates scan results per scan request.
   - Previous scan results are stored and can be retrieved using their unique ID.
   - UART based communication with the CC1350.
   - HTTP requests API with the frontend.
3. **Frontend**:
   - HTML Web page with the following components:
     - Vuejs Framework (DOM manipulation and requests).
     - Highcharts js (Charts display).
   - HTTP requests API with the backend.
   - Initiate scan configuration and poll for the additive results.

Flow Description

Complete flow

a. Access client UI using a web browser VIA the address http://127.0.0.1:5002/static/index.html (assuming the server runs locally).

b. Fill the from with the requested configuration parameters.

c. Click scan button.

d. Post request is sent to http://127.0.0.1:5002/api/scan/ with the scan configuration

   i. Request parameters:

      1. Top frequency.
      2. Bottom frequency.
      3. Step size.
      4. Number of times to sample the frequencies in each interval.
      5. Time between scan intervals.
      6. Number of intervals.

   ii. Backend returns a unique UUID for the client to poll.

e. Backend initiates scans periodically (by the parameter) and aggregates results.

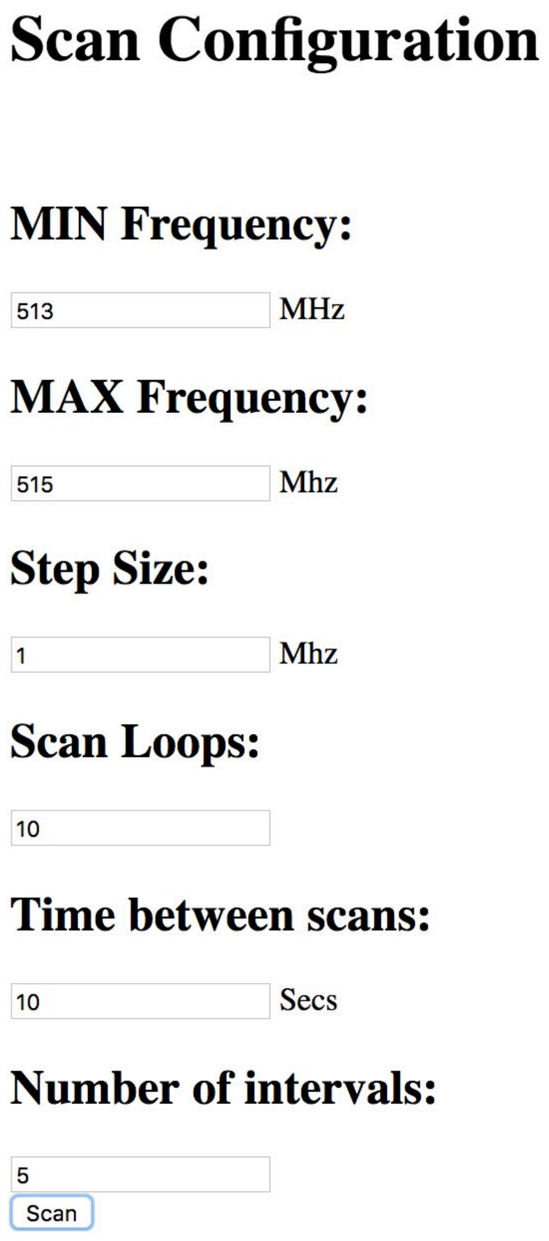
f. Frontend polls for the results on http://127.0.0.1:5002/api/results/ until backend turns on the 'done' flag.

   i. Request parameter is the scan UUID.

   ii. Every time results are back, charts are rendered again.

g. Results are presented both in raw form and aggregated (mean and median of each scan interval).

Scan Configuration Example



Scan Configuration

MIN Frequency:

513 MHz

MAX Frequency:

515 Mhz

Step Size:

1 Mhz

Scan Loops:

10

Time between scans:

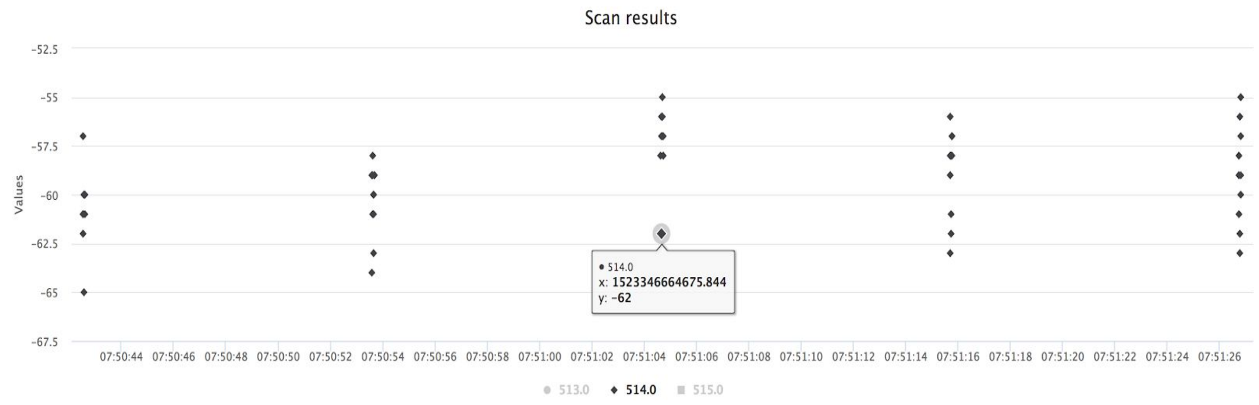
10 Secs

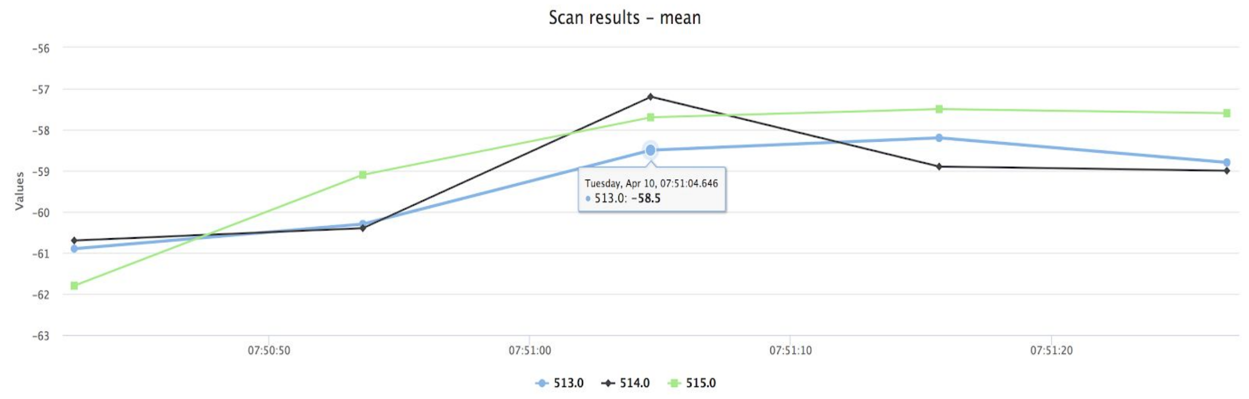
Number of intervals:

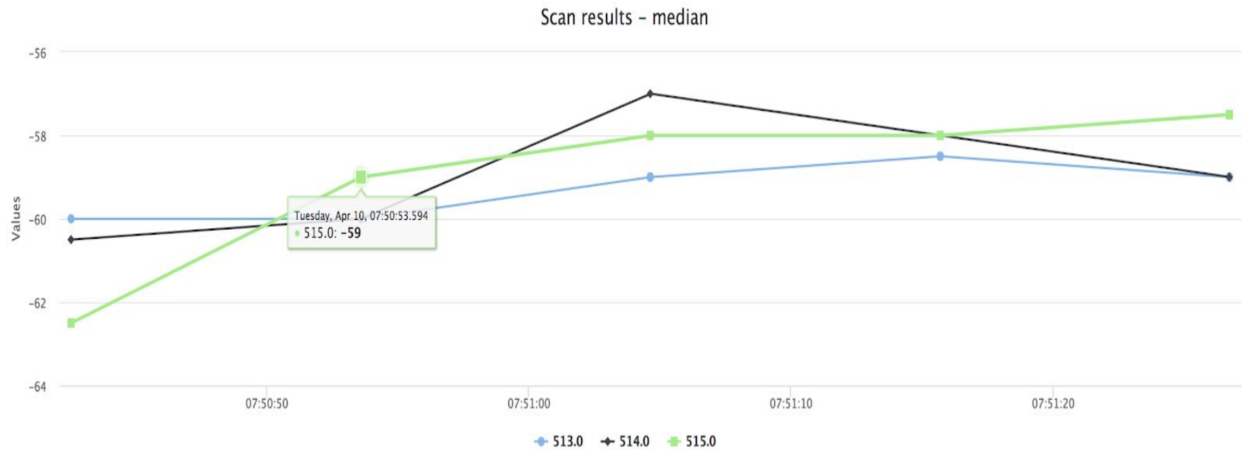
5

Apr 2018
Liad Wainberg 201324332
Assaf Cohen Shahar 304866163

Scan Results Example







Summary

The system enables a user to make and schedule RF scans and performs as we planned. In hindsight, there were several challenges we dealt with while building it:

a. Dynamically changing the frequency and getting a correct RSSI result once the frequency was changed.
b. Communicating with the board using UART and debugging its code while doing so (the CCS debugger and our server can't work together when using the serial connection).
c. Establishing an UART based communication protocol between the server and the board.

Further improvements to the system that could be added in the future in order to make it a more whole product:

a. Our server saves previous scan results and configurations, in order for this data to be non-volatile the server should use a DB to store and retrieve the data.
b. Adding client support for accessing older scan results.
c. Establish a different communication method, preferably wireless or through the internet, with the board that will support storing scan results on the board itself and extracting them when needed (thus enabling the board to work on batteries while disconnected from the server).