AS260-VCU-V1 Vehicle Control Unit

Quick Start guide for AS260-VCU-V1 Vehicle Control Unit

The quick start guide for AS260-VCU-V1 serves a guide for user to get start with AS260-VCU-V1 Vehicle control unit. It explains step by step procedure to program and debug the board and explains how to interface the board with external systems.

Contents

Table of Figures

1. Introduction

A Vehicle Control Unit (VCU) is one of the core ECUs (Electronics Control Unit) of the Automotive E/E architecture. AS260-VCU-V1 is a Vehicle control unit solution that interfaces with various ECUs via CAN, LIN and Ethernet ports. It also has sensor interfaces like SENT, PSI, I2C, SPI and a communication card connector to connect with cloud enabling connected vehicle technology. Figure 1-1 shows the block diagram of the solution.

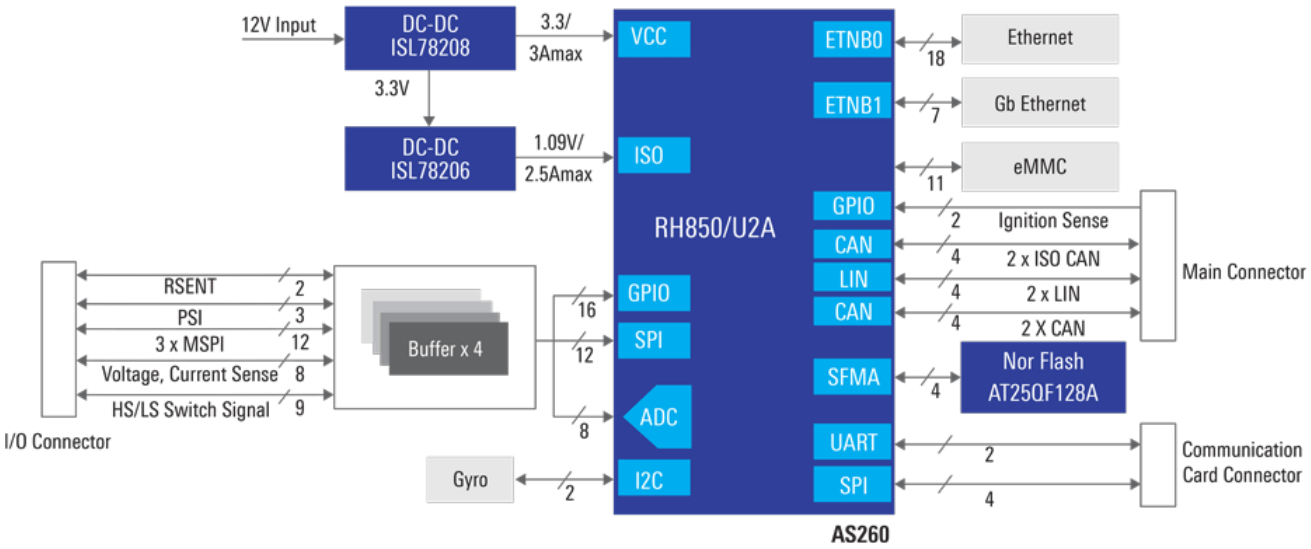

Figure 1-1 Vehicle control Unit Block Diagram

2. Scope of the document

The scope of the document is to explain the AS260-VCU-V1 Vehicle Control unit (VCU) hardware and guide the user to get started on the same with step-by-step procedure on programming, debugging and connections. The document also covers testing of Sample software provided with the package.

3. AS260-VCU-V1 Hardware

The AS260-VCU-V1 hosts Renesas's RH850U2A8 microcontroller (MCU) and various peripherals as shown in Figure 1-1. The MCU interfaces with external world through peripherals like CAN, LIN, SPI, UART etc. and various connectors provided on the board. Refer the figures Figure 3-1 and Figure 3-2, that show the connectors used to interface MCU to other system.

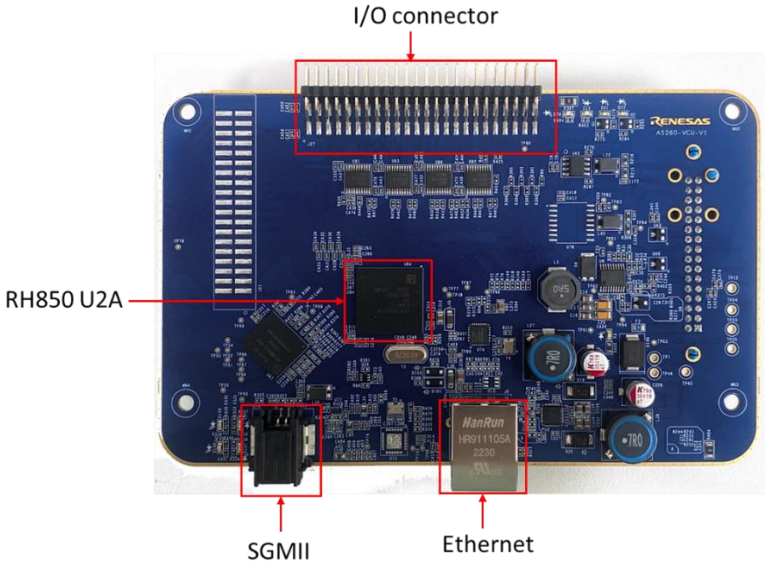

Figure 3-1. Front View of AS260-VCU-V1

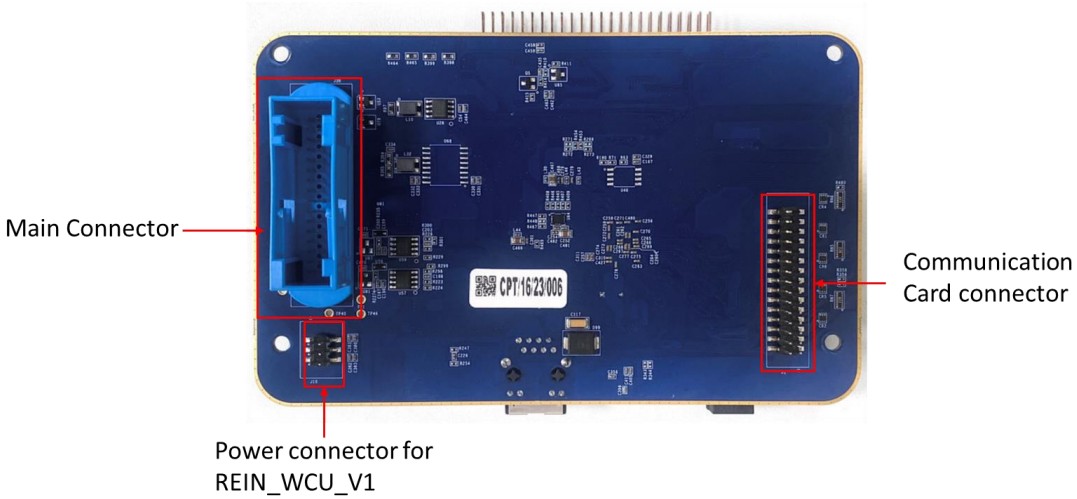

Figure 3-2 Back View

1. **I/O Connector:**
   The IO connector includes the SPI (SPI0, SPI1, SPI3 & SPI4), Eight Analog Inputs, I2C, RSENT, PSI5 and GPIO's. Thus, various sensors can be interfaced via this connector. Also, this connector is compatible with companion board of AS261 scalable BMS solution thus allowing board to board connection.

2. **Ethernet Ports**:
   AS260-VCU-V1 provides both 100 mbps and Gb ethernet ports which can be used for software upgrades, diagnostics, high data rate transfers and communicate with central processing units in zonal E/E architectures.

3. **Main Connector:**

   This is an automotive connector which has all main interfaces required for the board. Below is the list of interfaces on main connector:

   - Power supply
   - Debug interface
   - Four CAN interfaces.
   - Two LIN interfaces
   - Ignition sense

4. **Wireless Card Connector:**

   This is an expansion connector provided to connect to wireless modules such as Wifi, BLE, LTE over UART/SPI. Using this connector AS260-VCU-V1 can be interfaced with REIN _WCU_V1 (Wireless communication unit) solution.

5. **Power Connector for REIN_WCU_V1:**

   This is a power output connector that provides 3.3v and 3.8v output. The connector is mainly designed to power REIN_WCU_V1.

4. Getting Started with AS260-VCU-V1

This section provides the guidance on getting started with AS260-VCU-V1. Using the connectors shown in section AS260-VCU-V1 board, this board can be interfaced with other system and programmed using E2 debugger.

4.1 Programming and Debugging

As mentioned in section AS260-VCU-V1 Hardware the board hosts Main connector which has programming and debugging interface for MCU. The MCU, RH850U2A can be programmed using E2 emulator.

This microcontroller supports the following debug interfaces:

(1) Nexus JTAG Interface

(2) Low Pin Count Debug Interface (4-pin) (LPD4).

(3) In U2A-EVA emulation devices, a 4-lane Aurora trace port is provided.

(4) Trigger Input / Output pin (EVTI/EVTO)

The board comes with mating connector and wire harness which has a 14-pin debug connector as shown in below figure.

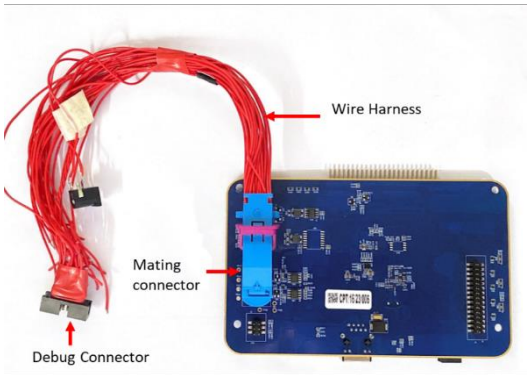

Figure 4-1 Mating connector with wire harness and debug connector

Below figure shows the connection of the board with E2 emulator.

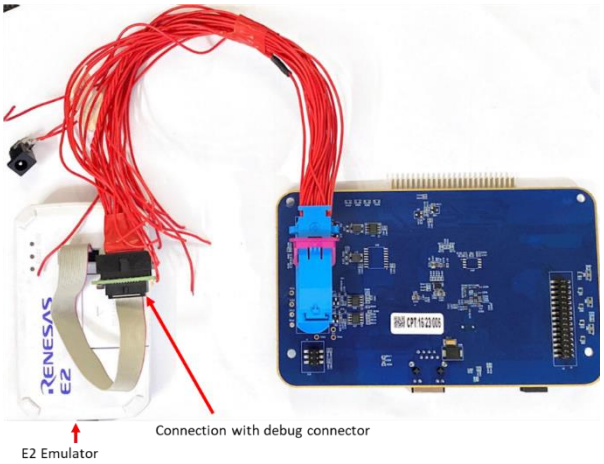

Figure 4-2 Connection with E2 emulator

4.1.1. Programming using RFP:

Renesas provides Renesas Flash programmer (RFP) tool to program RH850U2A device. Below are the steps to program U2A device using RFP:

1. Make the connections as shown in Figure 4-2. Power the board using power lines one harness.
2. Open RFP tool and navigate to ‘File’->New Project.

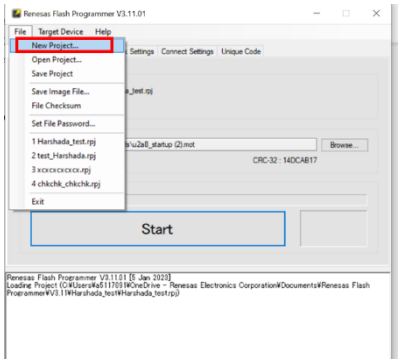

Figure 4-3 New project in RFP

3. Select the Microcontroller and enter the project name. Select the project folder and click on ‘Tool Details’

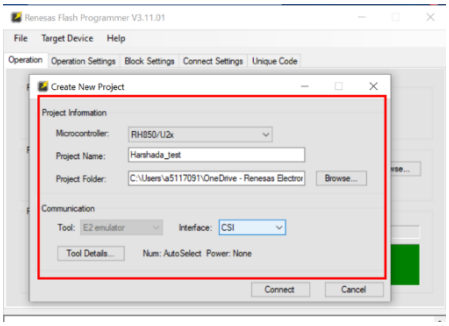

Figure 4-4 Creating new project.

4. Select Power supply as None.

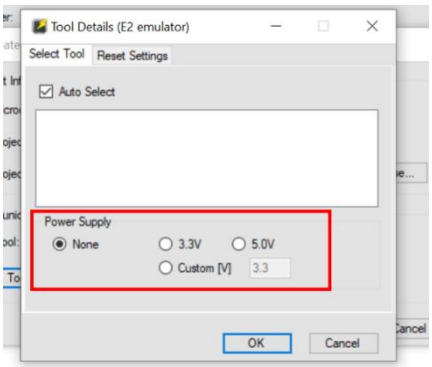

Figure 4-5 Select power supply.

5. Click on ‘Browse’ button and navigate to the desired ‘xxx.mot’ file which should be programmed in the board. Select the program file that needs to be programmed in the MCU. Note that, in case of multi-core project select the .mot file from the ‘DefaultBuild_merged’ folder of the project.

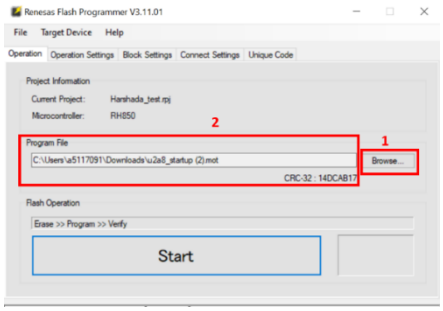

Figure 4-6 Browsing the project file

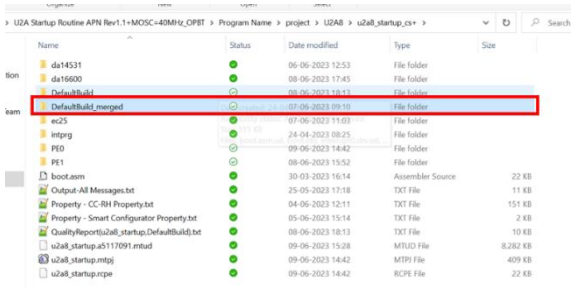

Figure 4-7 Selecting the .mot file from DefaultBuild_merged folder.

6. Navigate to Operation Settings and check the box for Erase, Program and Verify as shown below.

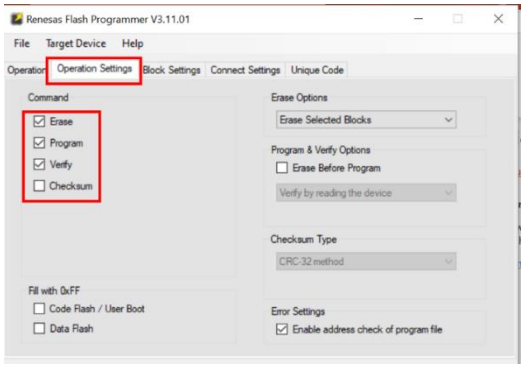

Figure 4-8 Operation settings

Navigate to Block Settings and select all the blocks shown below:

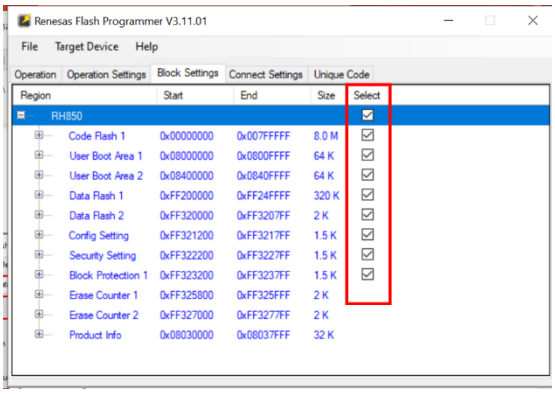

Figure 4-9 Block Settings

7. Navigate to ‘Connect Settings’ and select the settings as below:
    1. Select connect Settings.
    2. Select Interface for programming (2 wire UART/CSI).
    3. Keep the default speeds selected.
    4. Set the Main clock frequency as 20 MHz.
    5. Uncheck the override SVR parameters.

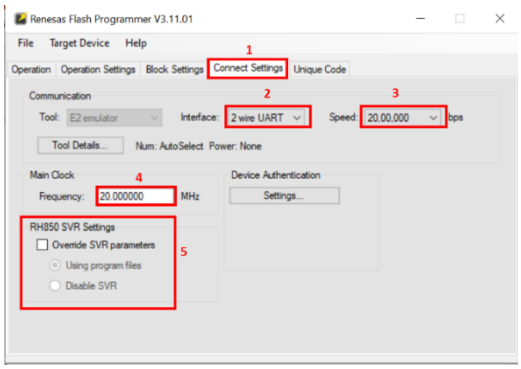

Figure 4-10 Connect Settings configuration.

8. Go back to operation and click start.

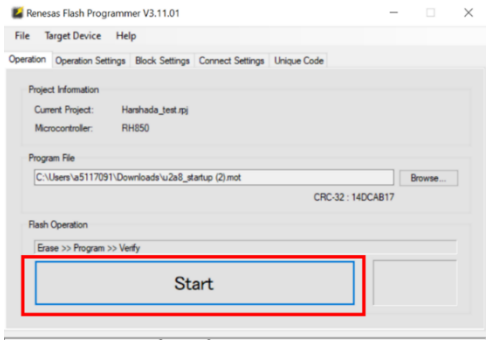

Figure 4-11 Start the Programming

9. The tool checks for authentication Code. Click OK and proceed.

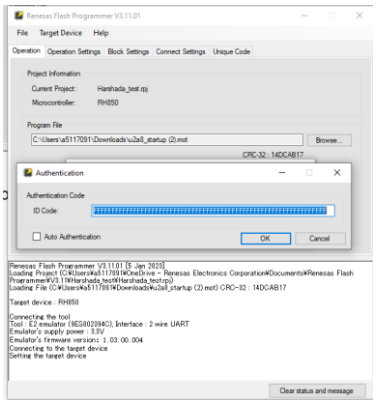

Figure 4-12 Authentication window

10. Click start button and the tool will show the progress of programming. Once programming is completed the tool shows ‘Operation Complete’ message as below:

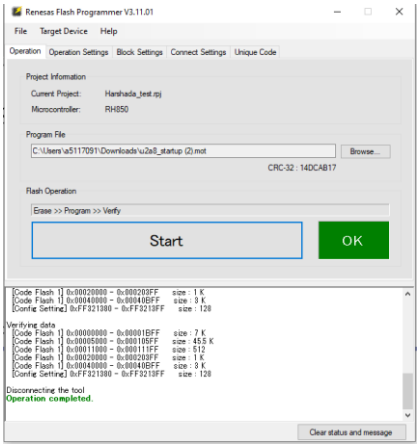

Figure 4-13 Programming progress

4.1.2. Debugging using CS+

RH850U2A can be programmed and debugged using CS+. Below are the steps to debug RH850U2A using CS+.

Open the RH850U2A8 project in CS+ by double clicking the xxx.mtpj.

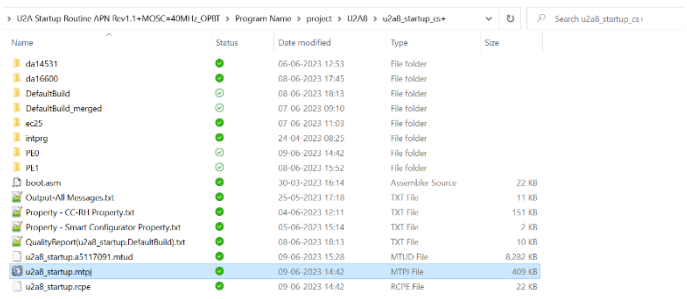

Figure 4-14 .mtpj file in the project

Once the project opens, click on build icon (marked as 1 in below figure), or press F7 to build the project. Once build is successful, click on download icon (marked as 2 I below figure), or press F6 to download the project.

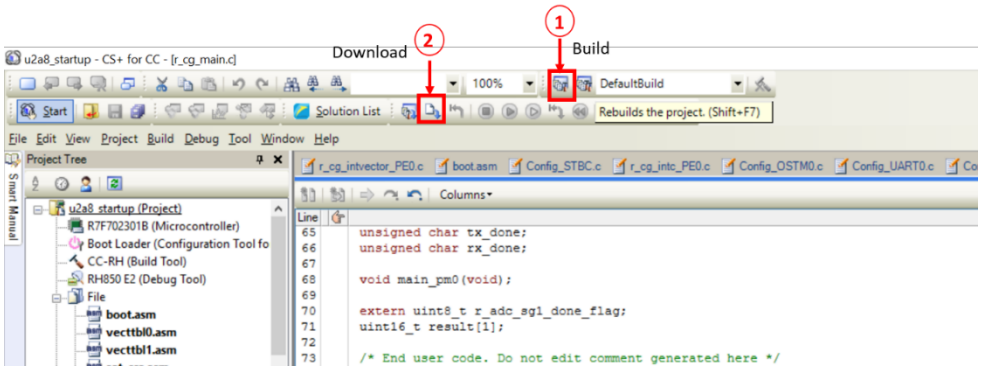

Figure 4-15 Build and Download the project

Once download is successful CS+ will open debug window as below:

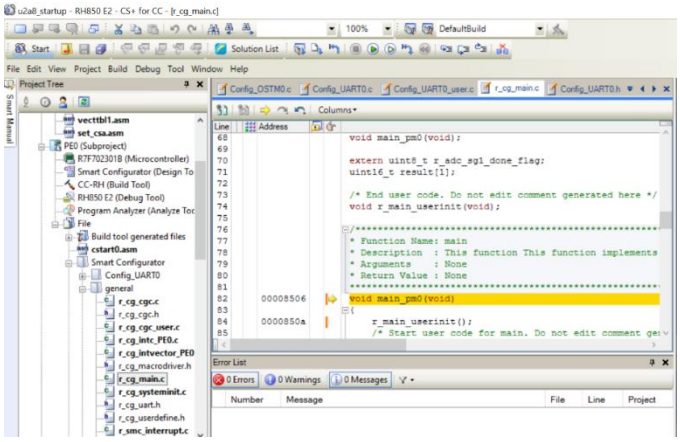

Figure 4-16 Debug window

To proceed debugging user can step in or step over by clicking the icons highlighted below:

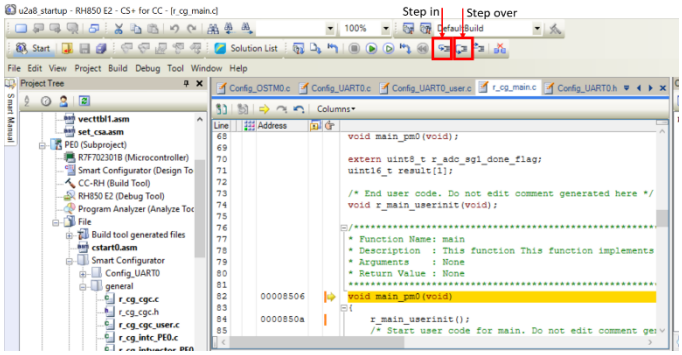

Figure 4-17 Step in and Step over options.

For more details refer below reference links:

[CS+ Quick Start Guide (4/4) - Debug | Renesas]

[CS+ | Renesas]

4.2 Interfacing with different system

As mentioned earlier, AS260-VCU-V1 can be interfaced with different system using the connectors provided on the board. Below sections give the details of the same.

4.2.1. Using I/O Connector

Below figure shows the schematic of the I/O connector. The MCU port pins are routed to I/O connector via buffer to add flexibility with the external system that operate at voltage other than 3.3V. VCCB_1, VCCB_2, VCCB_3 and VCCB_4 power supplies to the buffers. User should power these buffers at the desired voltages. For eg. The companion board operates at 3.3V/ 5V that has provision to power these buffers at 3.3V /5V.

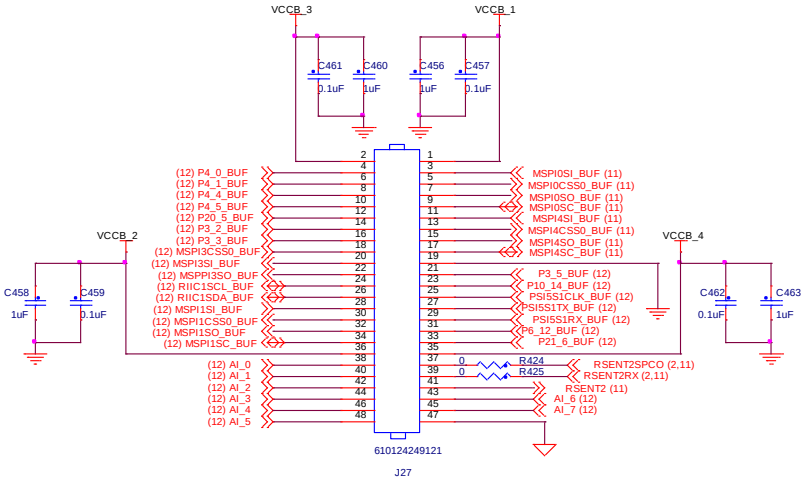

Figure 4-18 I/O connector

4.2.2. Using Main Connector

The main connector has power input, debugging interface, CAN and LIN interfaces. Below figure shows the schematic of main connector (J30).

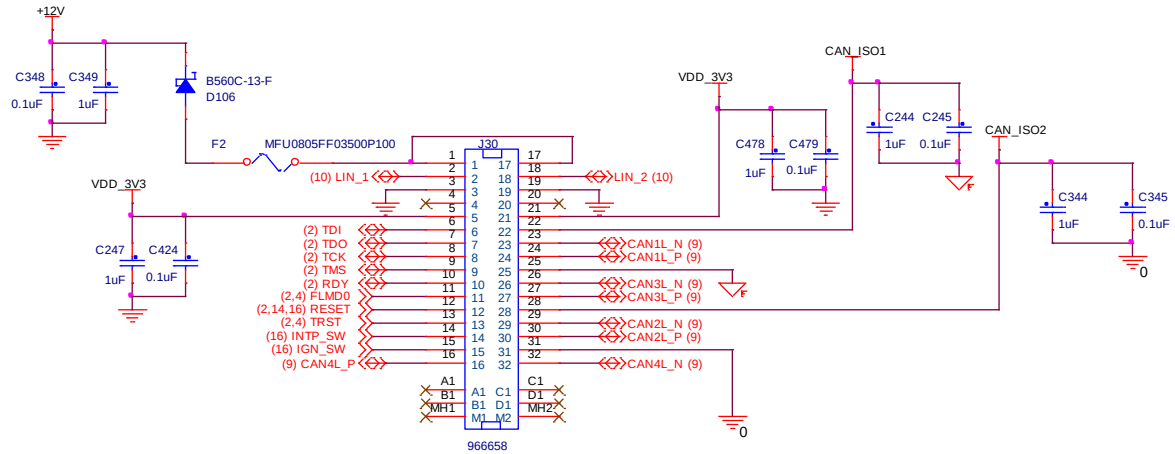

Figure 4-19 Main connector

Below table shows all the connections of the connector:

Pin No	Pin name	Signal Type	Pin description
1	+12V	Power input	Device 12V power input
2	LIN_1	LIN_1 bus	LIN interface
3	GND	12V Supply ground	Device ground
4	NC	NC	NC
5	+3.3V	Power output	Device +3.3V output
6	TDI	JTAG	Serial data input pin
7	TDO	JTAG	Serial data output pin
8	TCK	JTAG	Serial data input/output clock pin
9	TMS	JTAG	Mode select input pin
10	RDY	JTAG	Ready output
11	FLMD0	JTAG	Operation mode select pin
12	RESET	JTAG	Terminal reset
13	TRST	JTAG	Reset input pin
14	INTP_SW	Digital input	Intp switch input
15	IGN_SW	Digital input	Ignition switch input
16	CAN4L_P	CAN bus	CAN4_H interface of CAN-FD
17	+12V	Power input	Device 12V power input
18	LIN_2	LIN_2 bus	LIN interface
19	GND	12V Supply ground	Device ground
20	NC	NC	NC
21	+3.3V	Power output	Device +3.3V output
22	CAN_ISO1	Isolated 5V input	External Isolated CAN-FD 5V power input for “U68”
23	CAN1L_N	CAN bus	CAN1_L interface of Isolated CAN-FD

24	CAN1L_P	CAN bus	CAN1_H interface of isolated CAN-FD
25	CAN_GND	Isolated Ground	External Isolated CAN-FD ground for “U68”
26	CAN3L_N	CAN bus	CAN3_L interface of CAN-FD
27	CAN3L_P	CAN bus	CAN3_H interface of CAN-FD
28	CAN_ISO1	Isolated 5V input	External Isolated CAN-FD 5V power input for “U78”
29	CAN2L_N	CAN bus	CAN2_L interface of Isolated CAN-FD
30	CAN2L_P	CAN bus	CAN2_H interface of isolated CAN-FD
31	CAN_GND	Isolated Ground	External Isolated CAN-FD ground for “U78”
32	CAN4L_N	CAN bus	CAN4_L interface of CAN-FD

Table 1. Main Connector Pin

4.2.3. Interfacing with REIN_WCU_V1

To enable cloud connectivity AS260_WCU_V1 can be interfaced with REIN_WCU_V1 communication card, which has WiFi, BLE and LTE modules. The Wireless connector mentioned in Figure 3-2 is used to interface the two boards. These boards are designed to stack up on each other. Below figure shows the interfacing of the two boards:

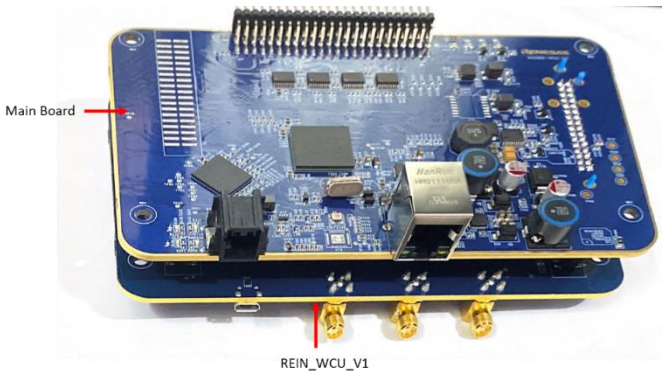

Figure 4-20 AS260+REIN_WCU

The boards are stacked up by connecting below jumpers to each other of the respective boards:

AS260_VCU_V1	REIN_WCU_V1
J1	J1
J10	J11

Table 2 Jumper connection to stack up the boards

4.2.4. Interfacing with AS261-Scalable BMS Solution

AS260_VCU_V1 can be interfaced with AS261 to form a complete BMS solution, where AS260-VCU-V1 performs all the BMS algorithm. Below figure shows the entire connection setup with AS260-VCU-V1 and AS261 boards.

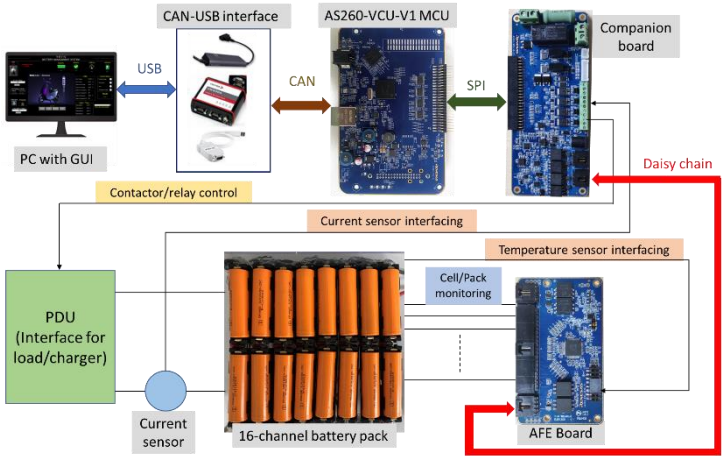

Figure 4-21 Complete BMS solution connection

AS260-VCU-V1 has I/O connector which is compatible with the board-to-board connector on companion board of AS261. Below figure shows the actual connection of AS260-VCU-V1 and companion board.

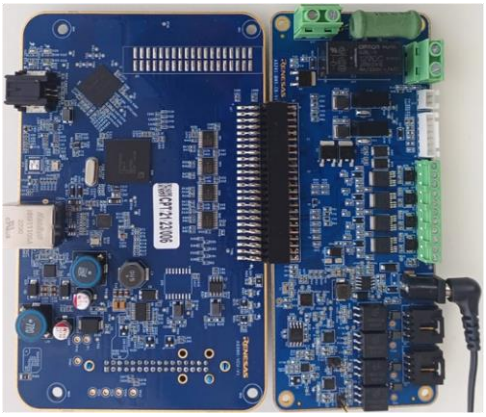

Figure 4-22 Connection with Companion board of AS261 BMS solution

For more details refer AS261 Scalable BMS user guide.

5. AS260-VCU-V1 Sample Software

The sample software has Battery Management solution algorithms and cloud connectivity over Wifi (DA16200), BLE (DA14531) and LTE(EC25) implemented on different cores of RH850/U2A8 microcontroller in AS260_VCU_V1. This should be tested with AS261-Scalable BMS solution and REIN-WCU-V1 referred in section 4.2.4 and 4.2.3 respectively.

The sample software performs calculation of BMS data for 16 cells and upload the relevant readings and faults to the cloud for further cloud computation. The BMS algorithm is done by core0, and cloud connectivity is done by core1 of RH850/U2A8 MCU. The data is shared between the cores using shared memory called Cluster RAM.

The Wifi module DA16200 on REIN-WCU-V1 is configured as HTTP client and the data is uploaded to HTTP python-based server. The BLE module DA14531, is BLE peripheral device and configured as serial profile to upload the data to the smartphone. The LTE module EC25, is connected to internet via SIM network and the data is uploaded over cloud over MQTT protocol. EC25 acts as MQTT client and is connected to “broker.mqttdashboard.com” with port 1883.

5.1 Prerequisites for testing the Sample Software.

Sample Software folder

Extract AS260VCUSamplesoftware.zip in the desired folder which has sample software project AS260_AS261_REIN_WCU_DualCore.zip and http server application folder ‘web’. The details of these are given in further sections.

Hardware

1. AS260-VCU-V1 board with harness
2. E2 emulator
3. AS261 Scalable BMS Companion Board (CB) and AFE boards.
4. REIN-WCU-V1 wireless connectivity board.
5. Battery pack

Software Tools

1. CS+, Renesas Flash programmer.
2. Python installed on PC: Python Releases for Windows | Python.org .

   Note that while installing the Python, you should check “Add python 3.10 to PATH” to add the environment variable to your PC, and select “Customize installation”, check the “pip” to install the pip together for the “flask” installation in the next step.

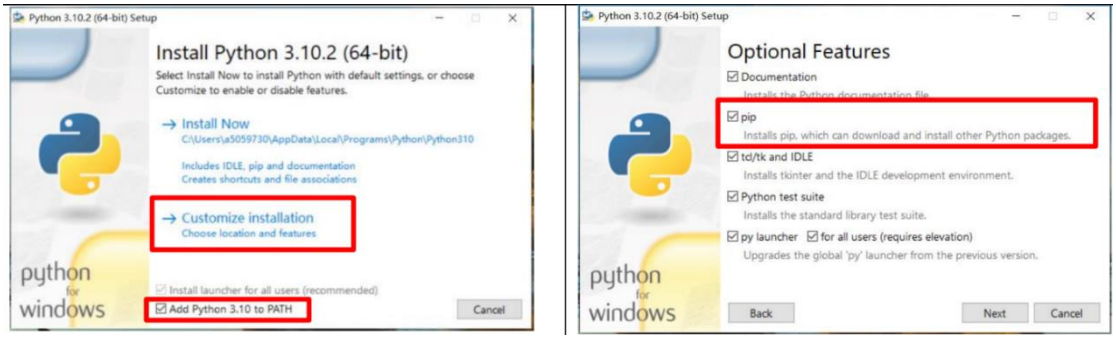

Figure 5-1 Installing Python

Open the “Command prompt” window, input “pip install flask” command to install flask. After installing it successfully, the following information will be shown.

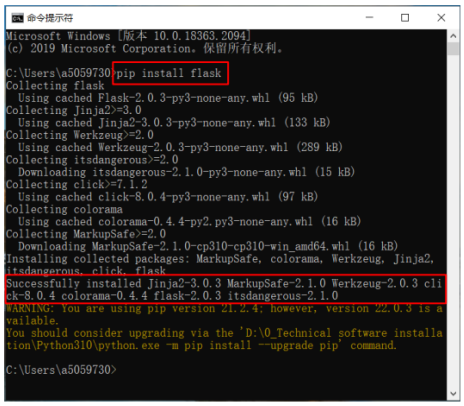

Figure 5-2 Installing flask

3. Smart Console mobile application installed on smart phone. Refer SmartConsole Android and iOS Application — DA145XX Tutorial SDK Getting started (renesas.com) for details on application download and usage.
4. Hivemq websocket MQTT client, MQTT Websocket Client (hivemq.com). Or any MQTT based client application connected to “broker.mqttdashboard.com” with port 1883.

5.2 Sample Software Folder Structure

Extract AS260_AS261_REIN_WCU_DualCore.zip in the desired folder to extract the sample software project. The project is based on CS+ platform compiled with CC-RH. Below is the folder structure of the sample software that shows two folders for each core PE0, PE1:

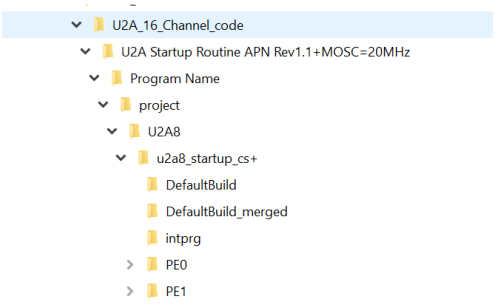

Figure 5-3 Top Project structure

Each core folder has relevant modules, software and src folder comprising of relevant drivers. PE0 folder has software files related to the BMS algorithm (ASW, BSW, INTERFACE_LAYER) and src folder with relevant drivers.

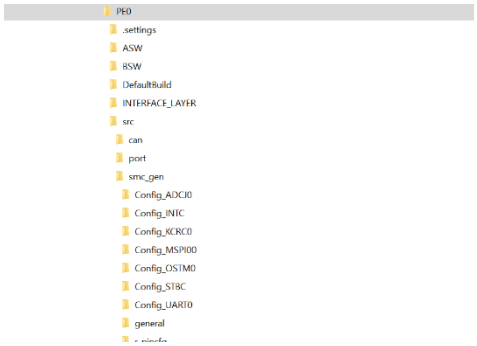

Figure 5-4 Core0 project structure

PE1 folder comprise of the modules related to Wifi, BLE and LTE connectivity and src files with relevant drivers:

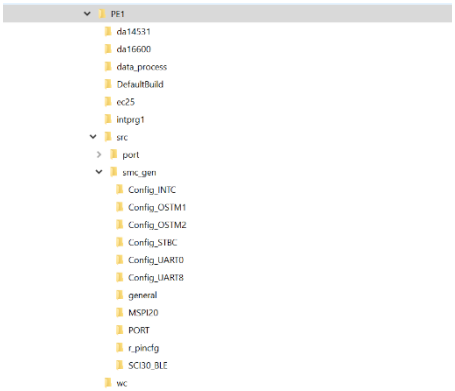

Figure 5-5 Core1 project Structure

5.3 Testing the Sample Software

To open the project, double click on u2a8_startup.mtpj in the below project path : U2A_16_Channel_code\U2A Startup Routine APN Rev1.1+MOSC=20MHz\Program Name\project\U2A8\u2a8_startup_cs+.

Once the project is opened you should set the server's name for HTTP connection in the project. The software package comes with for 'web' which has python application for HTTP server.

Locate the path to the "web" folder for HTTP server setup, for example, D:\web in Command Prompt. Input "httpserver.py" to run it. Then the HTTP server sets up. You can find the server IP address after the command run. For example, the HTTP server IP is http://192.168.8.21:5000 in the below figure. Press "Ctrl + C" to stop the HTTP server if need.

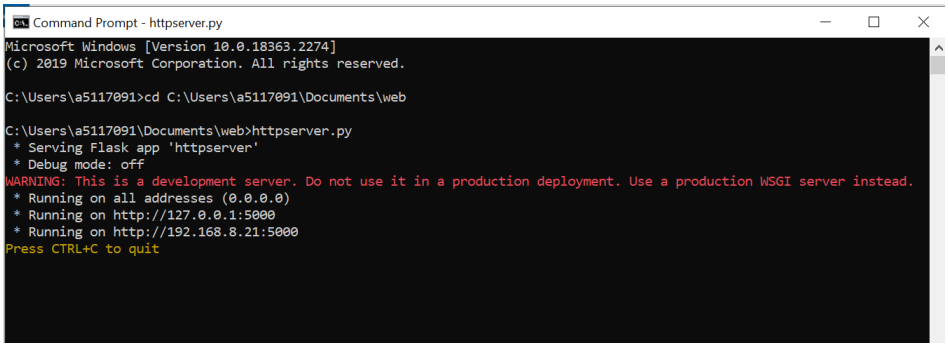

Figure 5-6 Run the HTTP server on PC.

You should set the IP address to the MCU source code, rebuild it, and then download it again to match your environment. In the project opened in CS+, navigate to hjp_wc.c in wc folder of PE1 core project. The edit the string “server_name[]” as per the server IP, as shown in the below figure and rebuild the application.

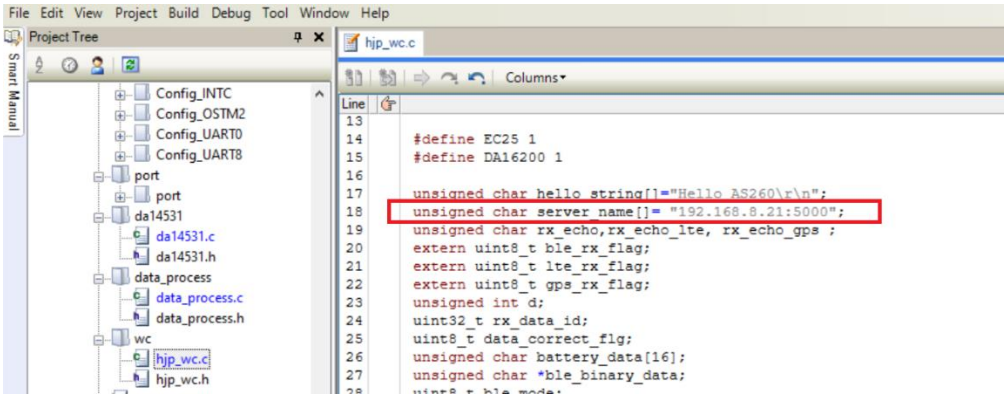

Figure 5-7 Edit the server IP

To program the board, follow the steps mentioned in section 4.1.1., program the ‘u2a8_startup.mot’ from DefaultBuild_merged folder.

For debugging follow the steps mentioned in the section 4.1.2.

Once programed, turn off the power, remove the E2 debugger and make the connections with AS261 and REIN-WCU-V1. Follow below steps while connecting the boards:

1. Connect AS260-VCU-V1 to CB board of AS261 using I/O connector as shown in Figure 4-22.
2. Stack up REIN-WCU-V1 to AS260-VCU-V1 using Wireless board connector as shown in Figure 4-20 Make sure that SIM card is inserted in SIM slot and, Main and Div antennas are connected on the REIN-WCU-V1 as shown in below figure. Also configure the SIM as per the network provider and country region.

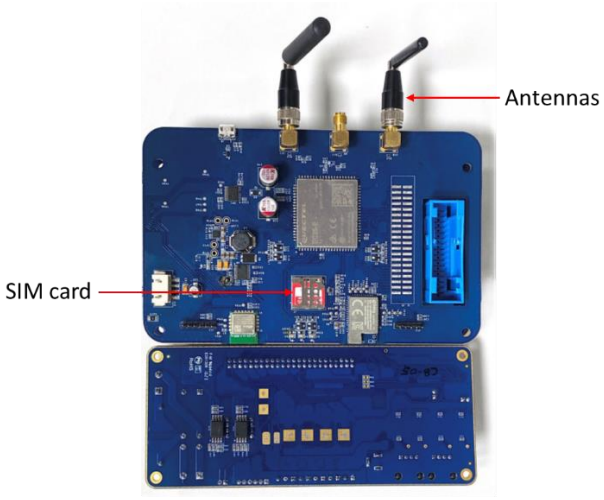

Figure 5-8. Antennas and SIM card on REIN_WCU_V1

3. Connect the AFE board and battery stack to CB.
4. Connect the 12V adaptor to each AS260-VCU-V1 and CB board.

Below figure depict the flow of the connections followed in entire system.

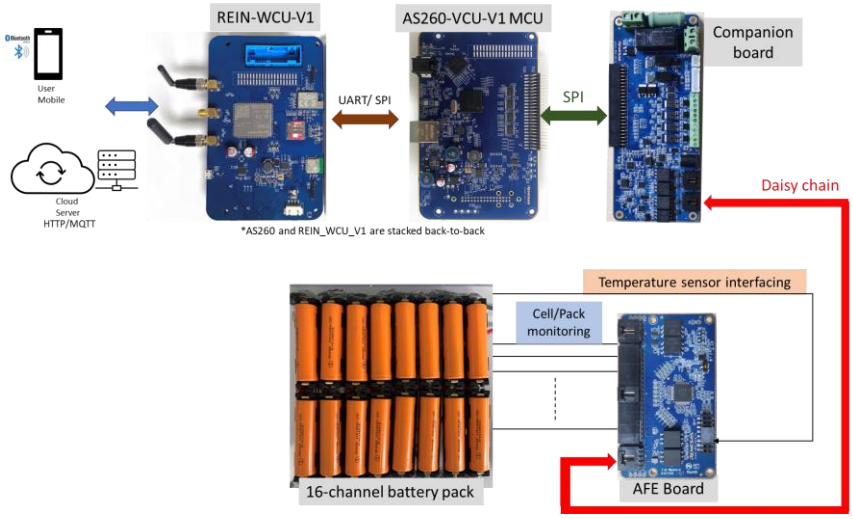

Figure 5-9 System Connection flow

Open the below applications on the respective devices for connecting respective modules:

1. Smart Console in smart phone.
2. Connect PC and DA16200 to same WiFi. Or turn on the hotspot function on PC, and then connect DA16200 to same hotspot.
3. Open HiveMQ WebSocket MQTT client on the browser.

Turn ON the boards. All the wireless modules will be initialized and be ready for the connection.

5.3.1. BLE (DA14531 module) Connection and Data Transfer

1. The Smart console will show ‘Clv2-Codeless’ in the list as shown below.

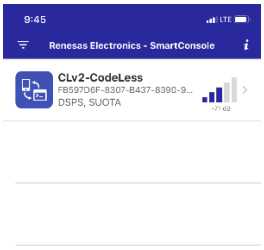

Figure 5-10. Smart Console detecting BLE device

2. Click it to connect to BLE device. The App will show Command mode window on it.

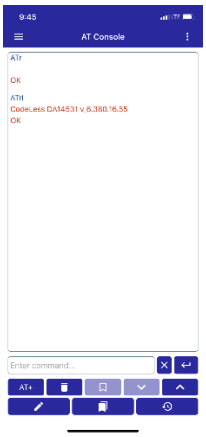

Figure 5-11. Command mode Window

3. After few seconds it will prompt Binary Request Received as shown below. Click on accept.

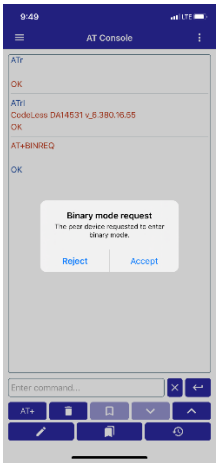

Figure 5-12. Binary mode request

4. After few seconds the data will appear as below on Smart console.

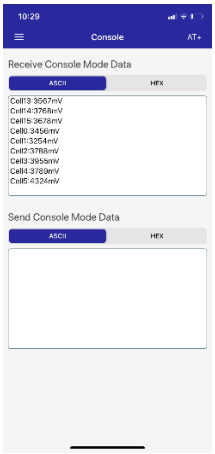

Figure 5-13. Data Reception on BLE

5.3.2. Data Upload over HTTP using Wifi Module

AS260-VCU-V1 communicates with DA16200 on REIN-WCU-V1 which acts as HTTP client and uploads the data to the PC based server. HTTP server will show the data as below.

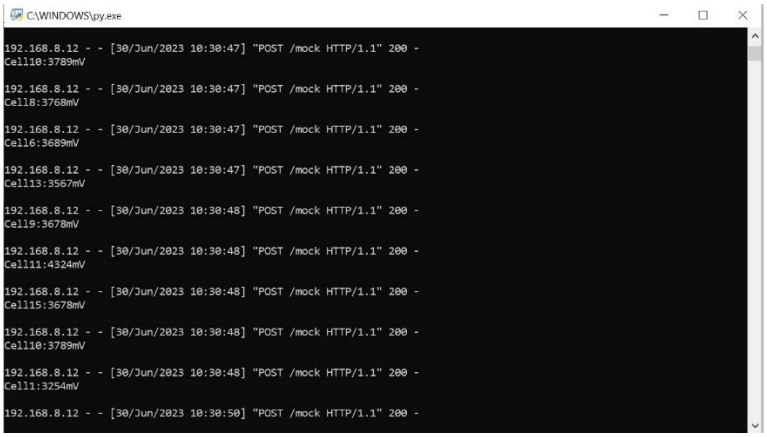

Figure 5-14. Data received on HTTP server

The data is display as ‘Celln : xxxxmV’, where n= cell number and xxxx is the cell voltage value in mV. For eg. If cell number 1 (n = 1) has 3254mV voltage (xxxx= 3254), the data displayed is as Cell1:3254mV.

5.3.3. Data Upload over MQTT using LTE Module

On HiveMQ client (or any MQTT client), connect to the broker by clicking on ‘Connect’ below:

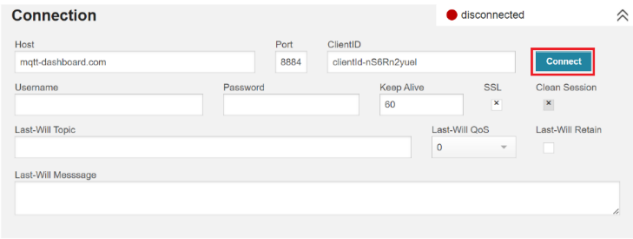

Figure 5-15 Connect HiveMQ client to the broker

Click on Add new subscription topic as shown below:

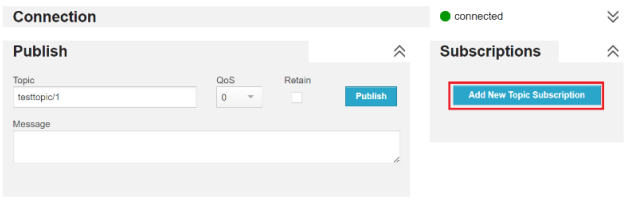

Figure 5-16. Add new subscription

Subscribe the topics, BMS/cell1, BMS/cell2…. BMS/cell16 as below.

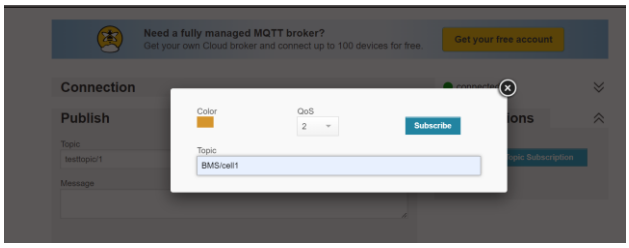

Figure 5-17 Subscribing topic in Hivemq WebSocket client.

Thus all 16-cell data will be uploaded and shown on the HiveMQ client. Below image shows six cell data uploaded in mV.

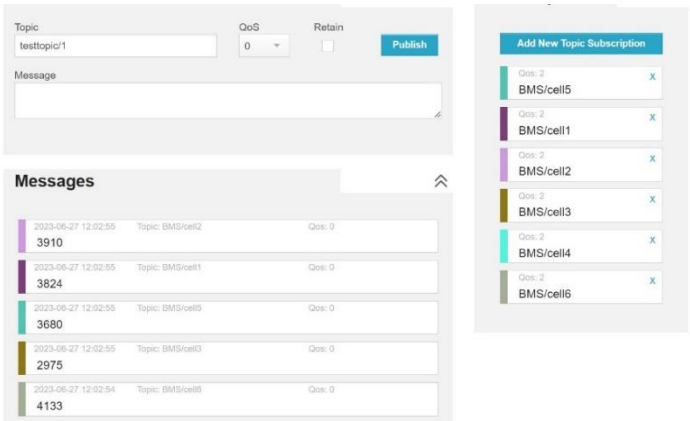

Figure 5-18. Cell Data uploaded over MQTT

Below is an example of IoT ON-OFF application on ios platform, which is MQTT client connected to “broker.mqttdashboard.com” with port 1883. This application shows graphical representation of the cell voltages received.

Figure 5-19. Graphical representation of Cell voltages vs time on IoT ON-OFF app

6. Acronyms and Abbreviations

Acronyms and Abbreviations	Explanation
BMS	Battery Management System
BLE	Bluetooth Low energy
HTTP	Hypertext Transfer Protocol
LTE	Long-Term Evolution
VCU	Vehicle control Unit
WCU	Wireless connectivity Unit
Wifi	Wireless Fidelity
MQTT	MQ Telemetry Transport

7. Revision History

Revision	Date	Description
1.00	June 30 2023	Initial release.