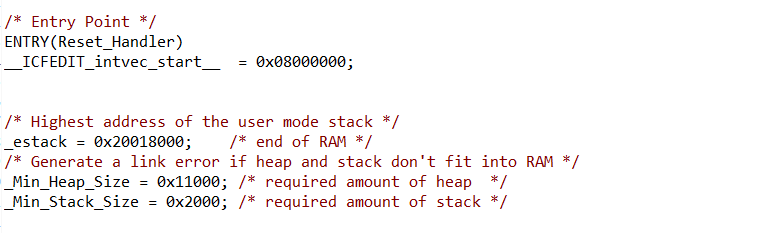
**GPS, CO Sensor & GCP Integration**

Summary of Tasks undertaken with problems listed at each point:

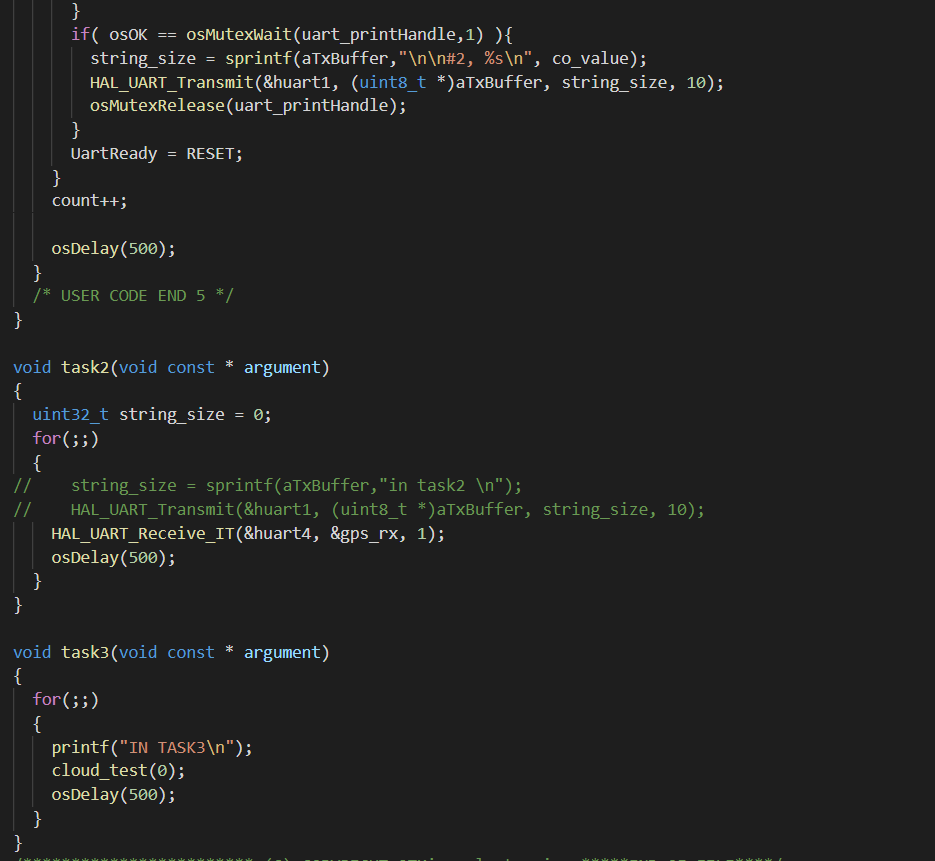
1. As per previous report, I had to add CO sensor with Google cloud and integrate it with FreeRTOS. One possible solution was changing the heap size from STM32L475VGTx\_FLASH.ld file in the project.



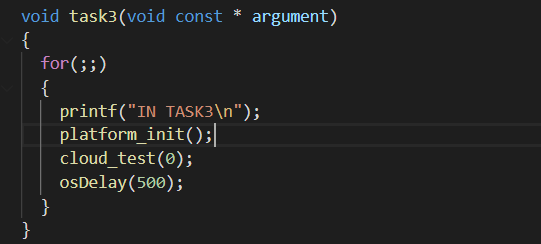
1. I changed the \_Min\_Heap\_Size from 0x11000 to 0x15000 did help, because heap is used by tasks, so we needed to change the heap size in FreeRTOSConfig.h. The variable name is configTOTAL\_HEAP\_SIZE.



1. Changing heap size from 3000 to 7500 helped executing 3 tasks. First task: GPS task, second: CO-Sensor task, third task: Cloud task. After setting the same delays, first and second task worked fine, but the third task did not execute at all. So I set the osDelay() as follows:



1. Placed the cloud\_test(0) in task and placed the platform\_init() before calling the scheduler in main().
2. Output was: **The output froze at Wifi initialisation**. Because the HAL\_GetSysTick() inside button function was not returning any ticks. Since the task scheduler didn't start, it didn't get any ticks and froze.
3. Placed the platform\_init() in the task. The terminal output was **net\_init() allocation failure.**

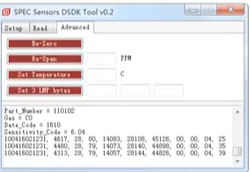


**SOLUTION TO THE ABOVE PROBLEMS:**

Hence, working with FreeRTOS did not work since GCP is still in working stage. So I choose to work on making CO Sensor work without FreeRTOS.

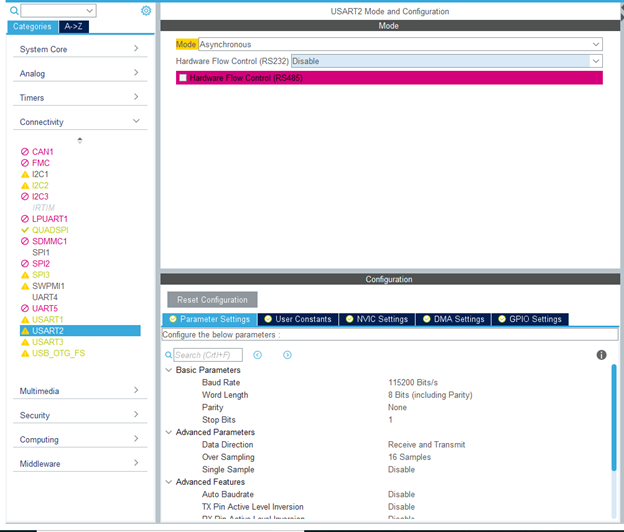
Approach 1: Working directly with CO-Sensor

1. As per DG-SDK-968-045 datasheet Page 7 (<https://www.spec-sensors.com/wp-content/uploads/2017/01/DG-SDK-968-045_9-6-17.pdf>), it is required to send a character “c” over UART for continuous data transmission from the sensor values.
2. First, I used a usb to ttl convertor and checked whether sensor was working. Got the proper output after sending “c” over the terminal.
3. Output format was: SN, PPB, T (°C), RH (%), ADC Raw, T Raw, RH Raw, Day, Hour, Minute, Second.



Approach 2: Directly working with HAL\_UART Interrupt API

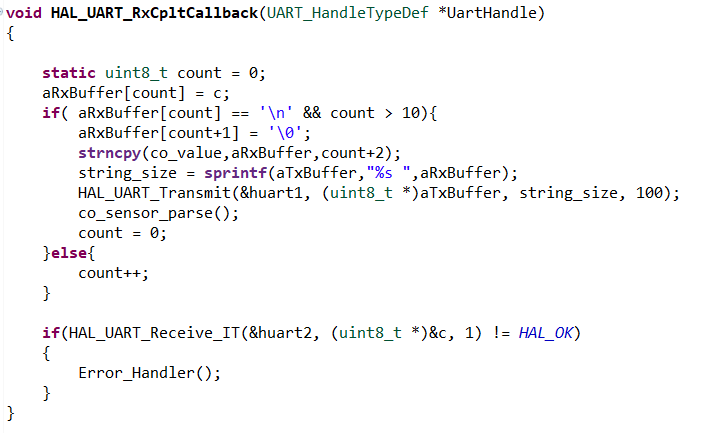
1. Opened CubeMX, selected the board, enabled UART2 at pin D4--RX and pin D10--TX. Enabled the UART2 global interrupt.



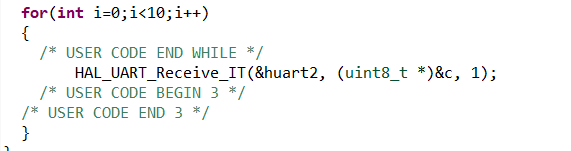
1. In code, I included the HAL Transmit API in main function to send the character ‘c’.



1. Included the HAL\_UART\_Receive\_IT() outside while(1) did not help since the output format contains 64 characters and everytime interrupt is triggered, it goes to UART\_RxCallBack() which includes algorithm to capture 64 characters and convert to string.



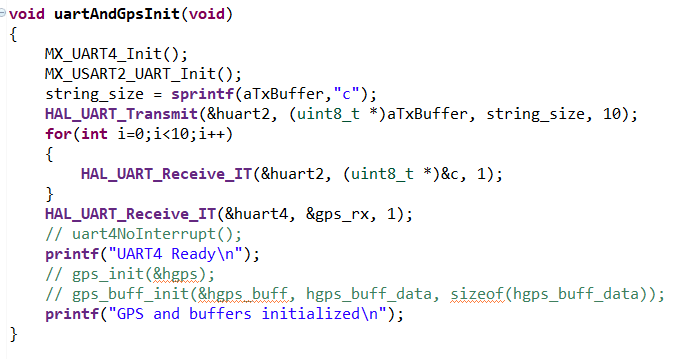
1. So placed the HAL\_UART\_Receive\_IT() with 10 for loop iterations and it started capturing data.



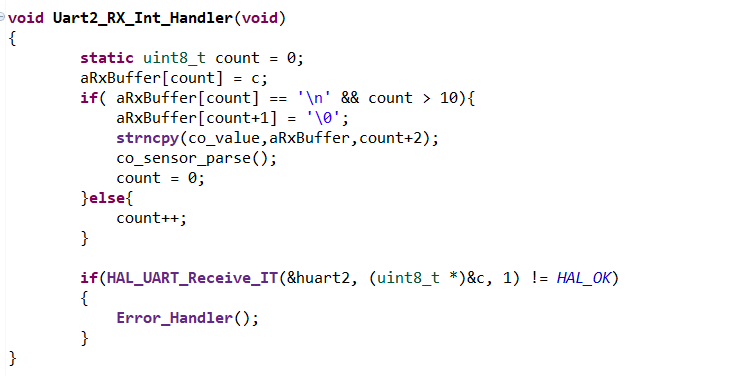
CO-Sensor was able to successfully execute without FreeRTOS. Next task was to integrate CO-sensor with GCP and GPS.

**Integration of CO-sensor and GPS code in GCP package**

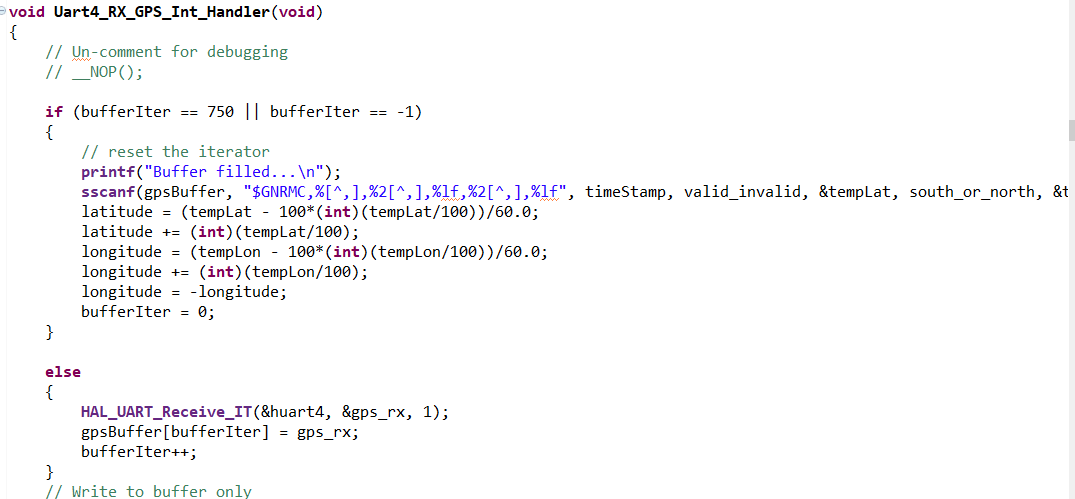
1. In googleiot.c, I added a init() which triggers UART2 for CO-Sensor and UART4 for GPS. I am calling the function before calling the gcp\_cloud function.



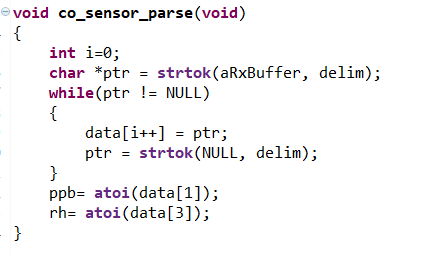
1. Once the gcp function starts, the UART2 and UART4 interrupts are triggered and UART\_RXCallback() was executed, where it started storing the sensor data as string.



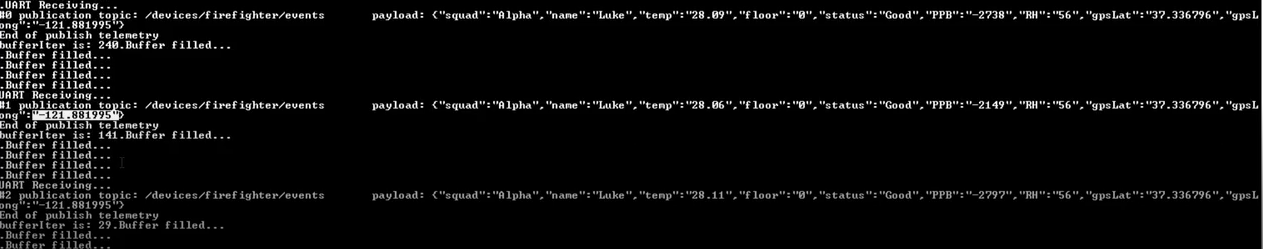
1. Since we are sending the data over JSON file, we had to parse GPS and CO-Sensor data. Worked with GCP code to parse GPS data. Commented parsing algorithm didn't work so I created my own parsing algorithm. Parsed lat and long successfully and appended the data in JSON file.
2. I used sscanf() to parse the data in form of string and convert to integer. Since the raw data contains latitude encode with time, I developed the algorithm as follows:



1. I used strtok() to parse the CO-Sensor Data(Parts per billion and Relative humidity). Since the data is separated by “,” , I used strtok() to scan the data and separate it from the main string. The algorithm is as follows:



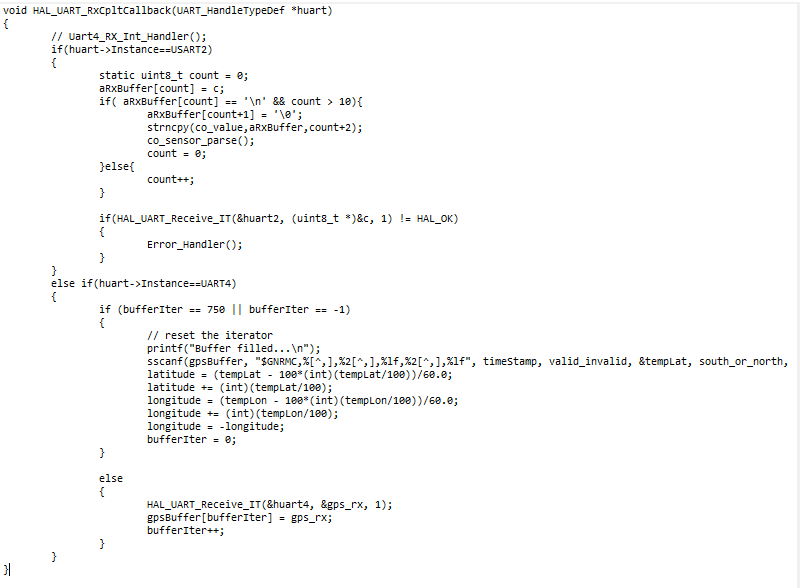
1. After running the terminal, configuring Wifi SSID, root CA and private certificates and google server address, we got the below output which is getting sent to the cloud with parsed GPS latitude, longitude and CO-Sensor Relative Humidity and Parts per billion data.



**Problem:**

**When I executed the GCP code, either GPS (UART4) data was going to the cloud or Co-Sensor (UART2) data was going to the cloud. Either lat, long=0 and Co-sensor publishes or ppb,rh=0 or GPS data publishes**

**Solution:**

The problem was interrupt callback was executed for one instance of UART in UART\_RXCallback(). 

Since interrupt routine should be as short as possible, I used 2 separate functions for UART2\_Callback() and UART4\_Callback() inside UART\_RxCallback() as follows:

