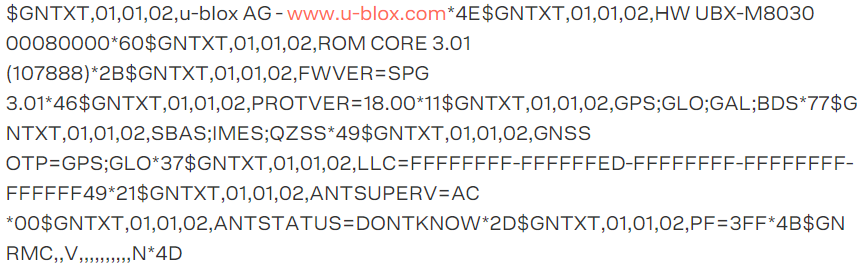
**Google IoT integration with FreeRTOS for sending GPS & CO-Sensor data to firebase**

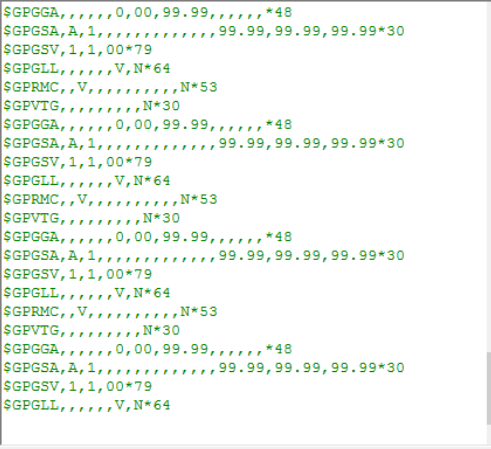
Date worked: 07/03 to 07/05 (15 hrs), 07/08 to 07/12 (24 hrs)

Summary of Tasks undertaken:

1. As per the last report, I was working on task synchronization for sending GPS and CO Sensor over the Google cloud with 3rd task implemented.
2. Task synchronization was successful after implementing proper OS delay with trial and error. CO Sensor requires a delay of 2000ms, GPS requires a delay of 500ms
3. Initially GPS\_task did not have any OS delay, which was creating problems with UART data at the output. I was constantly getting GNTXT messages as shown in the screenshot below:



1. After researching this particular problem online, the problem was **GPS data was filling up its RX buffer faster than it could parse, which was causing GPS module to reset**
2. I added the delay of 1000ms to GPS\_task and it was showing the output as per below screenshot:

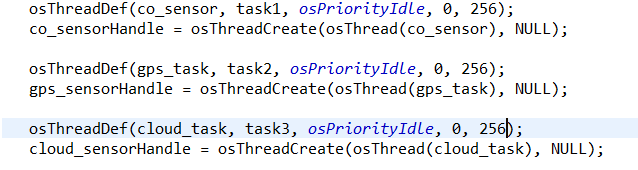


6. GPGGA and GPGSA did not give proper output. Now instead of faster, the data was slower. By trial and error, the delay was selected at 500ms **which took 3-4 days to figure out**

7. After setting the delay, task3 dedicated for google cloud setup was going into starvation. Meaning only two tasks for GPS and CO Sensor were context switching after each OS tick. **Hence, task 3 was never able to execute**

Possible solutions for task3 starvation:

* Reducing the delay for other task might solve the problem: After trying this solution, GPS task was getting affected and UART output from GPS was corrupted. Reducing the delay in CO sensor did not help, the code was sending character ‘C’ to sensor but in return, sensor was not sending data over UART2. **So changing the delay was not the correct solution.**
* After researching online for the next 2-3 days, I found one solution. THE HEAP SIZE DEDICATED FOR A TASK WAS MORE FOR GPS\_TASK AND CO\_SENSOR\_TASK. From the below code snippet, i had assigned **256 bytes** of stack size for all the 3 tasks.



Since OS works on preemptive scheduling and round robin algorithm, task1 executes-->fills up the heap size, task2 executes-->fills up the heap size, task3 comes from waiting queue→ ALL THE HEAP SIZE IS FILLED, hence it doesn’t get to execute and goes into starvation.

* Changing the stack size in the above code snippet to **128 bytes** solved the problem. Task3 started executing.

Summary of task undertaken(contd..)

8. After solving task3 starvation issue and adding a delay of 500ms, i ran the code and got the following the below output:



Net\_init() was not able to allocate the heap size of **struct net\_ckt\_t** using malloc() which caused returning NULL to the function. **Increasing the heap size in task3 osthread() did not help.**

9. I removed task3 and called the google cloud init function (cloud\_test(0)) inside the main(). As the below screenshot, the code was stuck and did not move further to configure WIFI



**Accomplishments: Google Cloud task was working with other 2 tasks with proper task synchronization and OS delay**

**Problems: I am still figuring out a way to allocate the heap size for google cloud task to make the net\_init() work. I am stuck with this problem for past couple of days now.**