C/C++ Software exercise

1 – Purpose

The goal of this test is to provide us with a full understanding of your coding style and skills. We'll pay particular attention to:

- The code structure
- Coding standard
- Choice of data structure
- Quality
- Reusability
- Use of unit testing
- Methodologies

The solution should not take more than two days to write, though it will be difficult to write a "complete" and top-notch solution in a short time span. The goal is not to get a solution covering all special cases in 100% robust way; the functions should be error free when used correctly but our main goal is to understand your approach to the problem.

2 – Description (requirements)

The autopilot CPU's runs a real time operating system (RTOS). It has been designed to have a task running at 200 Hz that is in charge of the reception of TSIP *packets* coming from the ground station. The RTOS tick frequency is 1000 Hz. This task has a priority that is not the highest one neither the lowest one.

The TSIP *packets* are received from a Serial Port. The Serial port is read using a DMA with a buffer size of 512 bytes.

The extended TSIP format protocol is used (Extended TSIP Format) with a maximum of 256 bytes TSIP *data* size.

The following function of the SDK is available to be used to collect telemetry raw data (containing TSIP *packets*) from the serial port. You do not need to program this function.

```
/**
 * \brief Read received raw data from COM port.
 *
 * This is a non-blocking read function. This means that only available received
 * data will be served. User may decide to call this function within a loop
 * until the desired amount of data is received.
 *
 * \param buf    Pointer to destination buffer where to copy received data.
 * \param count  Maximum number of bytes to be read.
 * \return       Number of read bytes. This value will always be <= count.
 */
int32_t uavnComRead(uint8_t * const buffer, const uint32_t count);
```

Once the TSIP *data* is extracted from TSIP *packet*, call the following function that will process such TSIP *data*. You can assume that this function consume the data performing the required autopilot actions. You do not need to program this function.

```
/**
 * \brief Parse a TSIP data.
 *
 * \param[in] buffer Const pointer where data is located.
 * \param[in] numberOfBytes It is filled with the number bytes in the TSIP packet.
 */
void ParseTsipData(const uint8_t * const buffer, const int32_t numberOfBytes);
```

3 – Exercise

Design and implement the needed code and the required tests to verify the correct behavior.
Deliver the code and a document with comments related to the design, code, tests, … before 1 week upon receiving the email. Do not hesitate asking questions if in doubts about the requirements.

```
void TaskUplink200Hz(void)
{
	// Fill this function. You may use additional functions if needed.
}
```

Tip: Split the exercise in two parts:

1. Create the TSIP parser to convert raw data (containing TSIP *packets*) into TSIP *data*.
2. Design the task flow using the TSIP parser.

Optional: The RTOS is configured as non-preemptive, but feel free to mention what would change if it were configured as preemptive.

Optional: The use of TDD is viewed favorably but it is not mandatory. If used, explain why you have used it in this scenario and its pros and cons.

Happy coding!