**SETR - Relatório da segunda tarefa**

**Docente:** Paulo Pedreiras

**Autores:** João Modesto, 97943

Luís Araújo, 93287

Como segunda tarefa no âmbito da UC – “Sistemas embutidos e de tempo real”, foi-nos pedido para implementar um módulo em C que processa comandos recebidos via UART, ou seja, entra um caracter de cada vez no buffer.

De forma a receber os valores emulados, é necessária uma simulação dos sensores. Foi criada a função “emulatesensors()” para conseguirmos obter valores perto dos reais. Esta parte de código, dependendo do 4º byte recebido, obtém valores do nosso sensor presente em Aveiro, no caso de ser ‘1’, bem como valores de sensores presentes noutras cidades como forma de trabalho extra:

* ‘1’- Aveiro, local;
* ‘2’ – Viseu, remoto;
* ‘3’ - Guarda, remoto;
* ‘4’ - Algarve, remoto;

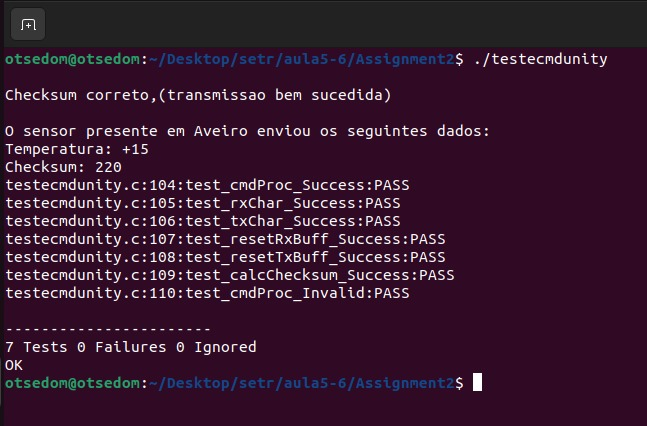
Para cada um dos casos anteriores, a temperatura, humidade e CO2 foram inicializados com valores médios de cada uma dessas regiões para a obtenção de valores perto dos reais, estes valores a cada chamada da função são levemente variados.

Antes da fase de testes do unity criámos o ficheiro “main.c”, que basicamente pede um comando ao utilizador pelo terminal onde podemos inserir por exemplo “#pt12071!”. Utilizando este comando, pedimos ao sensor para nos fornecer apenas o valor da temperatura utilizando a sequência de “pt”, o 4º byte como é ‘1’, sabemos que recebemos os valores locais, Aveiro, o 5º byte é responsável pela alteração da forma como os valores são mostrados. Utilizando o quinto byte a ‘1’, os valores são escritos no terminal tal e qual como estão no buffer, e usando a ‘2’ os valores são retornados de uma forma mais percetível e de mais fácil leitura.

O array de histórico vai atualizando os valores de temperatura, humidade e de CO2. Quando são guardados 20 valores de cada, os arrays são limpos do valor mais antigo para o mais recente.

Relativamente aos testes realizados, testámos a entrada de um caracter válido na entrada do buffer RX, no caso o ‘#’. Foi testado o cálculo do checksum de um comando de entrada válido ‘#pt12071!’ cuja soma dos bytes e posterior módulo de 256 é de 71, retornando “SUCCESS”. E por fim, a função principal de processamento. O cmdproc() foi testado para os casos seguintes:

* Receber um comando válido “#pt12071!” em que a sua resposta devolve o valor de temperatura proveniente dos sensores e o respetivo checksum.
* Receber um comando inválido “#ft42090!”, onde retorna *COMMAND\_INVALID*

O grupo teve o cuidado de testar a maior parte das variações de comandos que se podem enviar para o cmdproc(), com a finalidade de descobrir eventuais falhas ou erros, utilizando a “main.c “ onde é possível controlar o que entra no buffer TX através de strings captadas na consola previamente inseridas pelo utilizador.