classpert

Making Embedded Systems

Final Project Evaluation

Yellow Seahorses Cohort

Serial Snooper

by Can Caglar

Reviewers
André Araújo
Elecia White

1 Overview

The objective of this document is to assess and give you high-level feedback on your final project. Completing it and receiving a passing grade is a prerequisite for the certificate of course conclusion to be issued. Your project was reviewed and graded by mentor **André Araújo** and instructor **Elecia White**.

1.1 Project Details

Project Title
Serial Snooper

Student Name
Can Caglar

Enrollment ID
jhan_charler

Deliverables	Links
📄 Report	Open ↗
💻 Code	Open ↗
▶️ Video	Class Presentation Recording

2 Final Evaluation

For each criteria, a score was given according to the grading rubric (see appendix). The total achievable score was **24**, of which **18** are common credits and **6** are bonus credits.

Criteria	Score	Notes
Project meets minimum project goals	** breakdown 3.2	Interesting addition of TDD. Fully working CLI despite having few commands for the user.
Completeness of deliverables	**	Detailed report covering all project parts.
Clear intentions and working code	**	Well written and documented code.
Reusing code	**	Reused code was well referenced, including licenses (some were not explicit in the report, but could be easily obtained from the links).
Originality and scope of goals	**	Very interesting device with immediate application and opportunity for improvements.
Self-assessment (mentor category only)	**	Student score = 18, Mentor score = 18.5.
Power analysis, firmware update, or system profiling	**	Brief data rates analysis.
Version control was used	**	Very consistent Git usage with clear commit history.
	Total	** PASSED ▾

With a total of **, your project **PASSED**.

*Grades have been hidden

2.1 Reviewers' Feedback

2.2.1 Mentor Comments

by André Araújo

Hi Can!

This is an awesome project! Indeed, it is very useful to log data, both analog and digital, and having a simple tool to capture and store serial data for further analysis can be very convenient when designing and debugging embedded systems, especially when you work on devices with limited resources.

Sometimes we can't afford to have a good human interface on the system (DUT), but most certainly it will have a serial port available. Another situation, as you mentioned, is if there's a rare or intermittent bug and you may not be around the device exactly when it happens. So this device is really useful and something I'd want to have in my electronics toolbox.

Your report is complete and well laid out. I liked your drawings a lot! They're fun and also descriptive of the features you wanted to implement. Both hardware and software are well specified and explained, and it seems the project could be easily reproduced based on the information that was provided.

The DIP switch configurator is very practical for this kind of device. One future improvement you could explore would be trying to detect the DUT baud rate automatically.

Adding an OLED screen, as you suggested, is also a good idea to make the device more portable and friendly. If you're worried about memory consumption, maybe using only text (and not graphics) could save a bit of space.

I'm curious why you decided to use an external RTC, since the MCU has an integrated one. Using the internal RTC would be cheaper on the hardware side, and maybe simpler in software, also considering the millisecond counter you intend to add.

Having a battery is also necessary to keep time correctly. I know you plan to add battery support for the device to work, but have you thought about powering the device from the DUT and using a small battery just for timekeeping? Seems reasonable depending on the DUT and if you use a low power mode on the Snooper.

The TDD addition is very interesting. I have little experience with that but it's certainly something I plan to explore more in embedded projects, and your example will be very convenient!

Finally, you have developed a comprehensive command line interface that can be readily expanded. I liked how you divided software into several modules, which makes it easier to understand and maintain. Overall, your software is very well written and commented, good job!

Congratulations on this amazing project, Can! I'll make sure to keep an eye on your GitHub to see its evolution! Feel free to reach out on Discord/LinkedIn if you have any questions!

2.2.2 Instructors Comments

by Elecia White

André already mentioned the charming sketches in your report and the RTC needing a battery to retain time, even if you use a different power source for the Snooper.

I don't love the DIP switches. It was a good solution for now but you'll need to look up that table every time you modify the baud rate. Instead, I would put a file on the SD card with configuration parameters. It would let you add other parameters (whether to add the timestamp, use a different character to cause the timestamp, maximum file size).

I am so pleased by your report. You listened to me! And tried the things I suggested! Including tracking changes for optimizations! It is very exciting for me to see you use these tools. Adding in the extensive and strict Test Driven Development to your project means I got to see something I've never seen outside of a classroom. Neat!

Overall, great job. I am quite pleased to have met you and look forward to seeing more from you in the future.

3 Appendix

3.1 Grading Rubric

Criteria	Score		
	1 - Needs Improvement	**2 - Meets Expectation**	**3 - Exceeds Expectation**
Project meets minimum project goals	All project goals not met	All project goals are met. The state machine may be basic	Additional sensors, actuators Well documented and implemented state machine Comprehensive command line on serial port
Completeness of deliverables	Lacks report, video or code Report does not cover all sections Code has obvious errors that would cause it not to compile	Report covers all sections but some are answered incompletely leaving questions for the reader Code is readable given the report as a description Video shows code working	Code is readable on its own, without the report Report addresses each point thoroughly, demonstrating understanding as it related to the course Video demonstrates the project and is explanatory
Clear intentions and working code	What the system is supposed to do (based on the report or code) doesn't seem to be what the system does in the video	The system performs approximately as described in the report and code	The system performs as described in the report in a manner that is professionally polished The code shows how it works in a way that is easy for a maintainer to see

Reusing code	No code was used from other sources or it is unclear what code was used from other sources	Student code was identified	Versioning of reused code was included along with a license document that describes the license for the student's code and the reused code as well as shipping implications Reader is confident they could rebuild the student's system
Originality and scope of goals	The student did the bare minimum to meet the goals No originality	Some areas of interest were noted in the report but they were minor extensions of the existing examples	The student has gone far beyond the requirements to make something novel and awesome
Self-assessment (mentor category only)	Self-assessment was significantly different from mentor assessment	Self assessment was +/- 25% of mentor assessment	Self-assessment was +/- 10% of mentor assessment
Power analysis, firmware update, or system profiling	None	Described	Described, has graphs, and is accurate
Version control was used	None or a single commit		The log shows the project being built, though the messages may be terse but should be descriptive

3.2 Requirements

3.2.1 Project

🟢 Delivered 🟡 Partially Delivered 🔴 Not Delivered * Not Required ** Extra Credit

Features	Delivered	Note
Video turned in	🟢	Demoed live 11/19
Link to code	🟢	
Report turned in	🟢	
Use a Cortex-M processor	🟢	ST NUCLEO-F031K6 (STM32F031K6T6, Cortex-M0)
Button with interrupt	🟢	Not a button interrupt - UART triggers an interrupt
Has serial port output	🟢	Main feature of the system
Implements a state machine	🟢	CLI
Algorithmic piece	🟢	Circular buffer, CLI
Peripheral 1	🟢	UART (Snooper input and Console I/O)
Peripheral 2	🟢	SPI (SD card reader)
Peripheral 3	🟢	I2C (RTC)
Other*	🔴	
Other*	🔴	
Uses a HAL*	🟢	STM32Cube HAL and middleware (FatFs)
Analysis of Power**	🔴	
Firmware update**	🔴	
System Profiling**	🟡	Brief data rates analysis
Version control with history	🟢	Used Git with detailed commit history

3.2.2 Report

🟢 Delivered 🟡 Partially Delivered 🔴 Not Delivered

Features	Delivered	Note
Application Description	🟢	Clearly stated application
Hardware Description	🟢	Well explained
Software Description	🟢	Well explained software modules and interaction, including TDD
Identify written vs reused code	🟢	Reused code and relevant licenses identified
Architecture Diagrams	🟢	HW and SW diagrams
Build Instructions (HW)	🟢	Brief hardware list and connections table
Build Instructions (SW)	🟢	Building and testing instructions included
Debug Instructions	🟢	
Future Plans	🟢	Good ideas for future improvements
Self Assessment	🟢	