ECEN5623, Real-Time Systems:

Exercise #6 – Learning from Failure, Bettering Proposals

DUE: As Indicated on Canvas and Lecture

Please thoroughly read Chapters 11 & 12 in the main textbook.

This exercise asks you to explore some of the real-time software engineering errors made in historical systems and how they might have been avoided, particularly in your final project. In researching and reviewing the case, be sure to identify (or imagine) what the real-time requirements were and how failure to meet them resulted in failures – often facts will be unavailable, so do your best to reconstruct.

The group part of the exercise (in teams of 2, 3, 4 or 5) asks you to explore real-time software system analysis and design concepts by applying RM theory we have learned as well as software engineering skills you have learned as an Embedded Systems Engineering student. The goal is for your team to propose a real-time system that includes 2+ real-time services and requires RM analysis along with priority-preemptive (POSIX SCHED_FIFO) or RTOS priority scheduling to meet required periodic deadlines.

Exercise #6 Requirements:

1) [14 points] Research, identify and briefly **describe the 3 worst real-time mission critical designs errors** (and/or implementation errors) of all time [some candidates are Three Mile Island, Mars Observer, Ariane 5-501, Cluster spacecraft, Mars Climate Orbiter, ATT 4ESS Upgrade, Therac-25, Toyota ABS Software, Boeing MAX737] (These articles are online or in the zip on Canvas). Note that Apollo 11 and Mars Pathfinder had anomalies while on mission, but quick thinking and good design helped save those missions. State why the systems failed in terms of real-time requirements (deterministic or predictable response requirements) and if a real-time design error can be considered the root cause.

2) [16 points] The USS Yorktown is reported to have had a real-time interactive system failure after an upgrade to use a distributed Windows NT mission operations support system. Papers on the incident that left the USS Yorktown disabled at sea have been uploaded to Canvas for your review, but please also do your own research on the topic.
   a) [4 pts] Provide a **summary of key findings** by Gregory Slabodkin as reported in GCN. (Also available in the zip on Canvas)
   b) [4 pts] Can you find any papers written by other authors that disagree with the key findings of Gregory Slabodkin as reported in GCN? If so, what are the alternate key findings?

c) [4 pts] Based on your understanding, **describe what you believe to be the root cause** of the fatal and near fatal accidents involving this machine and whether the root cause at all involves the operator interface.

d) [4 pts] Do you believe that upgrade of the Aegis systems to RedHawk Linux or another variety of real-time Linux would help reduce operational anomalies and defects compared to adaptation of Windows or use of a traditional RTOS or Cyclic Executive? Please give at least 2 reasons why or why you would or would not recommend Linux.

3) [40 points] Form a team of 2, 3, 4, or 5 students and **propose a final Real-Time prototype**, experiment, or design exercise to do as a team. The proposal should either:

1) Provide a design and prototype of a real-time application with a number of Real-time services equal or more to the number of people in your team with deadlines,

2) Provide a design and prototype to address a current (contemporary) real-time application such as intelligent-transportation, UAV/UAS sense-and-avoid operations, interactive robotics, etc., or

3) Design an experiment to evaluate how well Linux on the Raspberry Pi or Jetson provides predictable response for real-time applications with at least one interrupt driven sensor and an observable output (e.g. audio, video display, actuation of a device).

For option #1, you could for example design and prototype a camera system to track a bright object (laser target designator) in a room, implement it, test it, and describe your design and potential improvements. For option #2, identify any contemporary emergent real-time application and prototype a specific feature (e.g. lane departure and steering correction for intelligent transportation), design, implement a prototype with the Rpi, DE1-SoC or Jetson, and describe a more complete real-time system design. For option #3, pick a test or simulated set of services (2 or more) which you can evaluate with Cheddar, with your own analysis, and then test with tracing and profiling to determine how well Linux really works for predictable response. You will be expected to write your own requirements for any of the 3 options. You will still need to provide a report which meets requirements outlined in Exercise #6 and make a presentation for your final exam, but the exact format should be tailored to your creative analysis, design and implementation.

a) [20 pts] Your Group should **submit a proposal** that outlines your exercise in real-time systems design and prototyping/testing with all group members clearly identified. This should include some research with citations (at least 3) or papers read, key methods to be used (from book references), and what you read and consulted.

b) [20 pts] **Each individual should turn in a paragraph on their role** in the project and an outline of what they intend to contribute.

Overall, provide a well-documented professional report of your findings, output, and tests so that it is easy for a colleague (or instructor) to understand what you've done. Include any C/C++ source code you write (or modify) and Makefiles needed to build your code. I will look at your report first, so it must be well written and clearly address each problem providing clear and concise responses to receive credit. Note that it is up to you to pick a programming language for implementation and tools of your choice to develop and test your application – this is a large part of the challenge of this assignment.

4) [10 points] **Provide all major functional capability requirements** for your real-time software system [not just for the proof-of-concept aspect to be demonstrated and analyzed, but the whole design concept envisioned]. You should have at least 5 major requirements or more. Requirements are not implementation details, but quantitative or qualitative descriptions of the performance of product needed to achieve its primary function. Please provide this in your report for the final project as well as your Exercise 6 submission.

5) [10 points] **Complete a high-level real-time software system functional description** and proposal along with a **single page block diagram** showing major elements (hardware and software) in your system (example) (example also on Canvas). You must include a single page block diagram, but also back this up with more details in corresponding CFD/DFD, state-machine, ERD and flow-chart diagrams as you see fit (**2 more minimum**). Please provide this in your report for the final project as well as your Exercise 6 submission.

6) [10 points] **Provide all major real-time service requirements with a description of each S_i including C_i, T_i, and D_i** with a description of how the request frequency and deadline was determined for each as well as how C_i was determined, estimated or measured as WCET for the service. Deadlines must be real and related to a published standard; or physical requirement based on mathematical modeling; or research paper or study. Please provide this in your report for the final project as well as your Exercise 6 submission.

Grading Rubric

[14 points] 3 worst real-time system failures (design and implementation flaws):

Section Evaluation	Points Possible	Score	Comments
System Failure #1	4		
System Failure #2	4		
System Failure #3	4		
Ranking	2		
Total	14		

[16 points] USS Yorktown investigation:

Section Evaluation	Points Possible	Score	Comments
Key findings on the USS Yorktown	4		
Alternate key findings	4		
Root cause analysis based on reading	4		
Would RedHawk Linux help for systems like the USS Yorktown?	4		
Total	16		

[40 points] Final exercise proposal:

Section Evaluation	Points Possible	Score	Comments
Group proposal	20		
Personal contribution	20		
Total	40		

[10 points] Functional (capability) requirements for system design:

Section Evaluation	Points Possible	Score	Comments
Requirement #1	2		
Requirement #2	2		
Requirement #3	2		
Requirement #4 to n	2		
Completeness	2		
Total	10		

[10 points] High level system and software design:

Section Evaluation	Points Possible	Score	Comments
Block diagram	4		
Detail diagram #1	3		
Detail diagram #2	3		
Total	10		

[10 points] Real-Time Services and Requirements:

Section Evaluation	Points Possible	Score	Comments
Services and descriptions	4		
C_i and WCET specification	3		
T_i and D_i specification	3		
Total	10		