Embedded Systems Application
B4M38AVS

Semester project proposal

Distributed embedded data acquisition system of an experimental rocket

Ondrej Nečas

Template: github.com francois-rozet sleek-template

April 27, 2024

Chapter 1

Motivation

The aim of the project is to create a C/C++ application designed as a distributed embedded DAQ (Data Acquisition) system that would simulate functionalities comparable to those found in a simple experimental rocket. Such a system would be able to monitor and process various environmental parameters, such as pressure or acceleration, and would on them provide real-time feedback to either itself or the operator.

However, due to the complex nature, extensiveness and the limited scope of the project, the proposed design focuses on to be an exemplary model of the real-world application. As the primary goal is, to illustrate the capabilities, only the basic concepts as distributed telemetry acquisition and transfer will be explored. Also, the emphasis is given on builtin visualization for the verification of the correct functionality.

To iterate, the main features of the proposed project are the data acquisition from various sources. Distributed retrieval and transfer of these sources across multiple processing units, and later visualization in a user-friendly manner, showcasing the system's ability to provide comprehensive real-time analysis of the surrounding environment.

1.1 Requirements

The proposed system fulfills the following minimal requirements:

- **Data Acquisition:** Telemetry collection from multiple peripheral devices.
- **Distributed processing:** Multiple units connected to a single communication network.
- **Data Visualization:** Graphical representation of the acquired data in real-time.

1.2 Project scope

The project core revolves around a main MCU (Microcontroller Unit) responsible for terminal data aggregation and visualization. Multiple secondary MCUs responsible for peripheral data acquisition, CAN (Controller Area Network) bus representing joint communication medium, underlying control software and final project documentation.

The pivotal role plays the acquired telemetry, which illustrates various environmental parameters. However, close to none additional processing is done on the acquired data as the primary focus will be on the handling of the data throughout the system and not its utilization. For the purpose of this project, the following data representatives are considered:

Data	**Description**	**Source**
Pressure	Measured in Pa	BMP
Temperature	Measured in °C	BMP
Acceleration	Measured in m/s^2 in G	9-DOF IMU
Rotation	Measured in degrees/s	9-DOF IMU
Magnetic field	Measured in microteslas (μT)	9-DOF IMU
Digital Inputs	Binary state	GPIO

Table 1.1. Acquired telemetry data

Terminal data aggregation is displayed to the end user/operator in a concise form. To correctly verify the system's functionality, the data should be visualized in a user-friendly manner. For this purpose, the main MCU has at its disposal multiple graphical peripherals. To name a few, an LCD (Liquid Crystal Display), 7-segment display, alphanumeric display and multiple LED (Light Emitting Diode) indicators are available. The built-in button and endpoint inputs also allow for the user to interact with the system and influence the final graphical output.

Software-wise, the project is dived into *layers*, *modules* and *libraries*. *Layers* represent the separation of concerns, where each layer is responsible for a specific task or function. On the other hand, *modules* represent the physical separation of the codebase for a specific MCU. *Libraries* are shared among the modules and layers. They provide the necessary functionality and allow for control with the underlying hardware.

Chapter 2

Implementation

2.1 System architecture

The architecture involves three separate MCU units. The main MCU (**STM32h747i-disco**) is responsible for the terminal data aggregation and visualization. Secondary and tertiary MCUs supplement the necessary telemetry. Sensor modules via the I2C (Inter-Integrated Circuit) bus provide all complex environmental data and to simulate any immediate interactions, GPIO (General Purpose Input Output) pins are included in the design. Subsequently, all the communication transfers between the MCUs are secured via the CAN bus that acts as a joint communication medium.

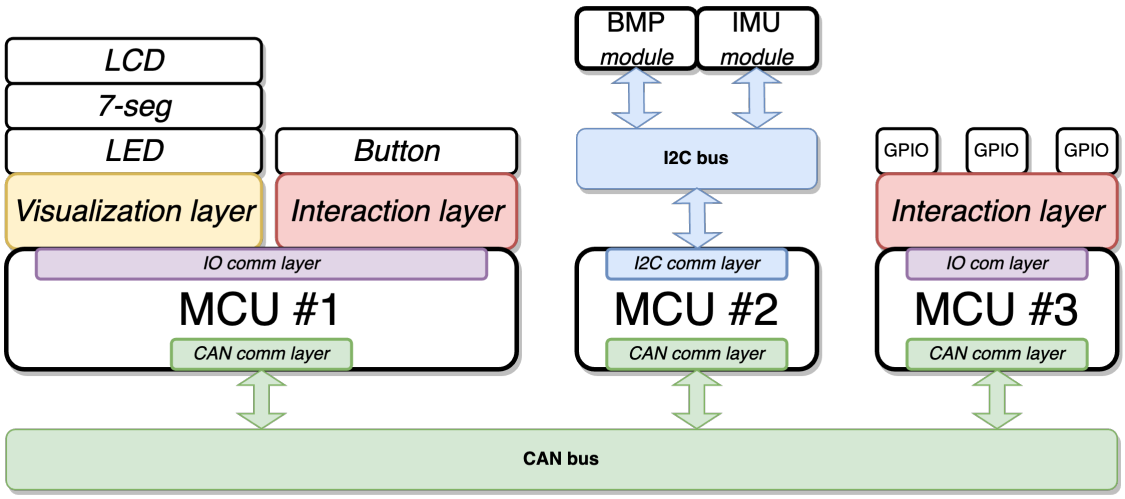

Figure 2.1. Top-level architecture diagram

From the data collection standpoint, the system is designed to acquire telemetry data on both periodic and immediate basis. Periodic collection is done by the sensor modules, and an immediate collection is done by the GPIO pins. This approach allows for a closer representation of the real-world application with various data sources.

To furthermore stratify the system, pivotal functionalities are divided into three layers.

Layer	Description
Communication	Data transfer
Interaction	User inputs
Visualization	Graphical outputs

Table 2.1. System functionality layers

2.2 Communication Layer

The communication layer consists of two segments.

Primary communication is done via the CAN bus on a multi-primary (also, multi-master) communication model. Messages regarding the GPIO data will have the highest priority as they will be responding to the system's immediate inputs. Messages with the sensor data will have lower priority and will be sent periodically. The initial frequency is set to 100Hz. And the CAN bus will be used to encompass all joint communication between the MCUs.

Secondary communication is done via the I2C bus and GPIO pins. Secondary MCU will use primary-secondary (also, master-slave) communication with the sensor modules through the I2C to simulate the environmental data. GPIO pins will be used for simpler digital input signals to simulate immediate interactions.

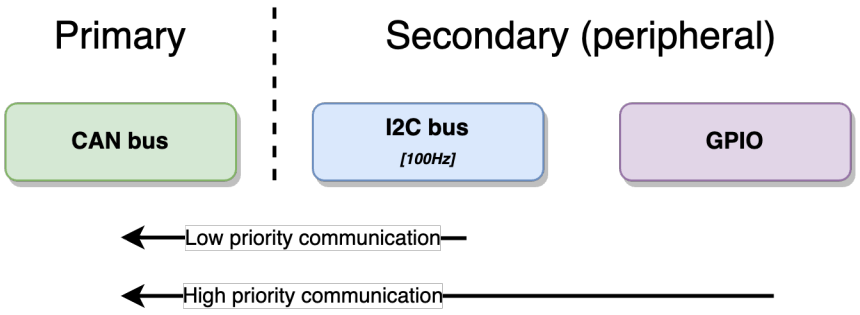

Figure 2.2. Communication layer segmentation

2.3 Interaction layer

To introduce the element of user control, the interaction layer is divided into two segments.

Navigation, the user can interact with the visualization subsystem via the built-in button. To switch between different presentation modes and settings.

Stimuli, the user can influence the system's state by providing digital input signals. By interacting with the GPIO pins, the system should immediately respond and simulate the critical or direct environmental changes.

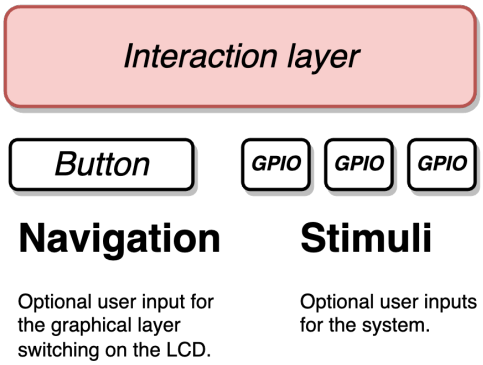

Figure 2.3. Interaction layer segmentation

2.4 Visualization layer

Final visualization is done in three segments.

Telemetry, presentation of the acquired data from the sensor modules. Concise summary of the measured values with the ability to switch between different data formats and visualization layers.

Navigation, control cue for the user of the data presentation. The user is informed about different presentation modes and the current setting.

Signaling, visual feedback of the system's state.

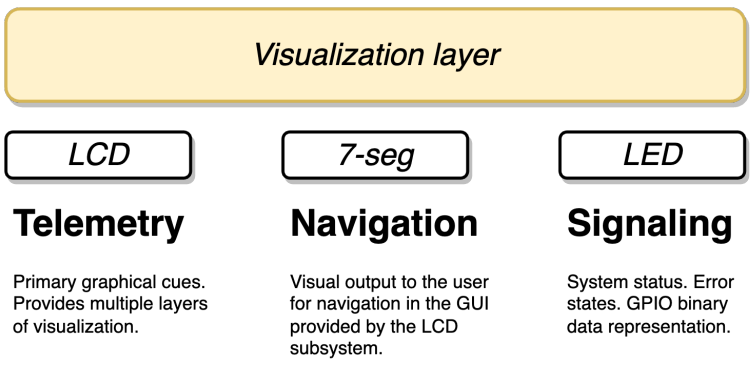

Figure 2.4. Visualization layer segmentation

Chapter 3

Summary

This document lays out the foundational principles, its architecture, and the communication model with its relations to the underlying hardware.

In a limited scope, the project aims to create a distributed embedded data acquisition system that would simulate functionalities comparable to those found in an experimental rocket. Apart from the real application, which would be much more complex and extensive, the proposed instalment is designed to be a small standalone unit that would serve as an exemplary model.

Given emphasis on exploring the capabilities of a distributed processing paradigm, the system is built around three MCUs connected via the single CAN bus representing the joint communication medium. On top of that, the system is designed to acquire telemetry data from various sources and lastly visualize it in a user-friendly manner to provide comprehensive real-time analysis.