Take Home Task

Role: Embedded Linux Software Engineer

Directions: While this task will take roughly 4-6 hours, you will have 48 hours to complete this task. This task is proprietary to Science. Do not distribute.

Science is excited to present you with this coding challenge as part of our interview process. This exercise will help us understand your coding style, problem-solving approach, and technical capabilities.

Problem Statement

When recording neural signals from the brain, minimizing interference from external noise is of utmost importance, while improvements in the analog electronics can reduce noise, some digital signal processing is always necessary.

Your task is to create a program that applies a notch filter that effectively removes a 60 Hz background signal from 256 channels of recorded neural data.
Your filter must be able to run in real-time and have minimal (>1ms) of delay between the input and output.

Example data has been provided in the form of a raw binary file containing samples in signed 16-bit format. This is unencoded binary data that can be viewed using a tool like xxd or the VSCode hex editor extension. The samples from 0-255 represent the first sample from each of the 256 channels, 256-511 the second sample for each channel, etc. The sample rate of the data is 32000 Hz. The data has been provided in the file “neural_data_256ch_16b.bin”

Functional Requirements

- Filter must run in real-time
- Filter must not introduce more than 1 millisecond of delay
- Program must parse the provided neural data recording and save processed data to the same format

Technical Requirements

You may use any programming language you are comfortable with to complete the challenge, although a compiled language is heavily preferred. Your code must compile and run on a standard linux system in order to be considered a valid solution.

You should use the following coefficients for a two-pole butterworth IIR filter:
b0 = 0.99901921
b1 = -1.99790074
b2 = 0.99901921
a0 = 1
a1 = -1.99790074
a2 = 0.99803843
These coefficients have been calculated using the scipy library. You must implement the core data-processing logic yourself.

If you make use of external tools, please include a description of what you used them for, and why.

What We're Looking For

- Well structured, readable code.
- Effective problem solving.
- Clear documentation.
- A performance-aware solution that scales well.
- Proper error handling.

Time Expectation

We respect your time and expect this challenge to take approximately 4-6 hours. Please don't spend more than 6 hours on this exercise.

Submission

Please include a file titled README to recruiting@science.xyz that outlines the following:

- How to compile your code, including any required software or environment setup.
- How to run your program.
- An explanation of how your filter works.
- Any potential shortcomings of your approach.
- Any assumptions you made.
- The performance profile of your code. Roughly how much memory does your filter use? What is the average compute time per sample?