**Filtering of CAN messages from an MDF (mf4) file**

Project request

Table of contents

[Introduction 3](#_Toc208485348)

[High-level requirements for MDF and CAN message filtering script 4](#_Toc208485349)

[Data details 5](#_Toc208485350)

[Provided inputs 6](#_Toc208485351)

# Introduction

In modern automotive and industrial environments, CAN (Controller Area Network) loggings often contain an extremely high volume of data - sometimes exceeding 20,000 signals per recording session. However, for most engineering, diagnostic, or data analysis tasks, only a small subset of these messages and signals are actually relevant for further handling and usage.

To address this challenge, the goal is to develop a script that efficiently filters CAN messages from an MDF (mf4) file, extracting only the messages and signals of interest. This targeted extraction not only streamlines subsequent data processing but also significantly reduces storage and computational overhead.

The following section outline the requirements and expectations for such a script, ensuring it is robust, maintainable, and ready for collaborative development within a version-controlled environment.

# High-level requirements for MDF and CAN message filtering script

1. The script shall use existing and available Python libraries and packages which provide CAN-bus and MDF support for Python.
   * *Hint: Investigate which Python libraries are available for reading and writing MDF (MF4) files and handling CAN messages.*
2. The filtering of CAN messages shall be based on a list of CAN message IDs found in the provided input DBC file.
   * *Hint: The script may support loading filter criteria from an external JSON file, but a static filter list is also acceptable. External definition of filtering will serve as a configuration which will further lead to not having a need to modify Python script every time a new message filtering is requested for different messages.*
3. The script shall read CAN messages from the input MDF file, apply the specified filters, and write only the matching messages to the output MDF file.
   * *Hint: If no filters are provided, all messages should be written to the output file.*
4. The script shall accept the paths to an input MDF (MF4) file containing CAN messages and to an output MDF (MF4) file where filtered messages will be saved.
5. The script shall provide a command-line interface to specify the input MDF file, output MDF file, and the JSON file containing filter criteria (if used).
6. The script shall handle missing or invalid input files, output files, and filter files gracefully, providing clear error messages to the user.
   * *Hint: It is fine to use print for this.*
7. The script shall include a descriptive header comment at the top of the file and docstring comments for all public functions, explaining their purpose, parameters, and usage.
8. The project shall be stored in a Git repository and a proper file structure must be created, including (but not limited to):
   * + A main script directory (e.g., src/)
     + A directory for input MDF and DBC files (e.g., inputs/)
     + A directory for outout MDF file(s) (e.g., outputs/)
     + A README file with usage instructions
     + A .gitignore file

# Data details

* Output data type: Filtered MDF file \*.mf4 (subset of CAN messages)
* Input data type: Vehicle MDF file (containing 20k+ signals)
* Required CAN messages (3 specific CAN messages):
* EngineData, ID: 0x64
* EngineDataIEEE, ID: 0x66
* Ignition\_Info, ID: 0x67

# Provided inputs

* MDF file: Logging\_MDF.mf4
* DBC file: PowerTrain\_MDF.dbc

The input files will be provided as a ZIP file.