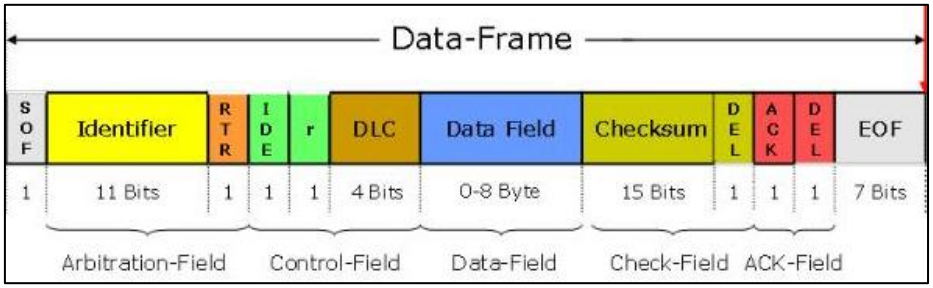
The CAN protocol is widely used in vehicles, to allow electronic control units (ECUs) to communicate over the vehicle's network bus.



You can further read of the CAN frame format in the following link:
https://en.wikipedia.org/wiki/CAN_bus#Data_frame

Please implement a system using Python which will be compounded of the following components:

1. **Generator Unit** – Will generate random traffic of CAN messages, corresponding to the following constraints:
    a. CAN frame ID – one of the following Identifiers (base format of 11 bits long): 0x100, 0x200 & 0x300.
    b. CAN frame DATA – random length of 0-8 bytes of data, of random data. Please notice that "DLC" field is the generated data length. **All other data fields should be constant "1".**
    c. A frame will be generated every random period time of 50mSec to 100 mSec.

2. **Detection Unit** – receives CAN traffic from the generator unit, and classifies it as ***valid*** or ***invalid*** as per the following criteria:
    a. ***Rate*** – a frame is classified as valid if it entered the unit later than 100mSec past the previous frame carrying the same ID (Otherwise Invalid).
    b. **Length** - a frame is classified as valid if its data length is different than the previous frame carrying the same ID (Otherwise Invalid).
    c. **Data** - a frame is classified as valid if none of its data bytes is contained in the previous frame carrying the same ID (Otherwise Invalid).

3. **Reporting Unit** – Receives input from the detection unit and writes the following fields (comma-separated) to a log file for each entry (entry = CAN frame):
    a. Frame's arrival timestamp (to detection unit).
    b. Frame (in Hexadecimal format)
    c. Valid / Invalid.
    d. If Invalid – for which reason (Rate/Length/Data).