**CHUYÊN ĐỀ THIẾT KẾ HỆ THỐNG NHÚNG 1**

**BÀI THỰC HÀNH 4: LẬP TRÌNH CAN NÂNG CAO**

# Mục tiêu

* Giúp sinh viên tìm hiểu và tích lũy kinh nghiệm trong việc chẩn đoán lỗi cho các bộ điều khiển điện tử (ECU – Electronic control unit) dùng cho các hệ thống ô tô cao cấp.
* Giúp sinh viên làm quen với chuẩn chẩn đoán UDS-ISO 14229 bằng việc hiện thực một số dịch vụ quan trọng ($27 – Security access; $22 - ReadDataByIdentifier và $2E – WriteDataByIdentifier).
* Giúp sinh viên tích lũy kinh nghiệm trong việc hiện thực lớp CAN\_TP ở tầng vận chuyển của giao thức CAN. Mục đích chính là giúp sinh viên có thể truyền/nhận gói dữ liệu với kích thước lớn hơn 8 byte)

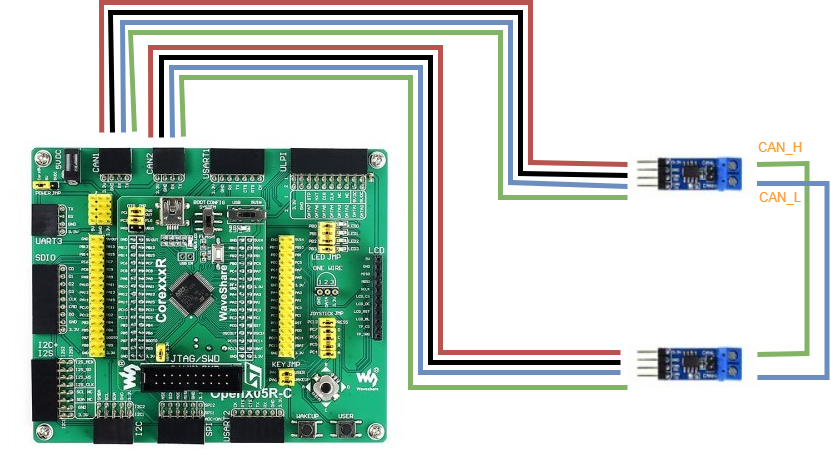
# Chuẩn bị trước

* Sinh viên xem lại Bosch CAN Specification 2.0

# Nội dung và hướng dẫn thực hành

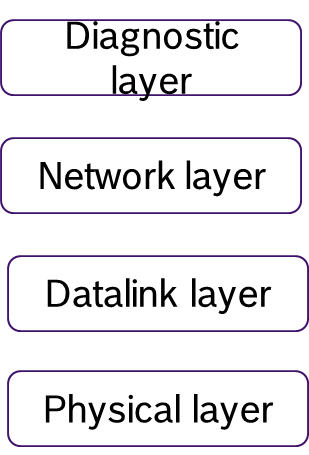
## Tổng quan hệ thống

Trên KIT thực hành có 2 module CAN1 và CAN2, sinh viên thực hiện kết nối mạng CAN giữa 2 module CAN1 và CAN 2 như **Hình 1**.



**Hình 1 Sơ đồ kết nối giữa CAN1 và CAN2**

CAN1 sẽ đóng vai trò là **Tester** module và CAN2 sẽ đóng vai trò là **ECU**. Hai module sẽ thực hiện giao tiếp chẩn đoán thông qua giao thức CAN. **Hình 2** mô tả bố trí các lớp trong giao thức CAN. Đặc điểm của mỗi lớp sẽ được trình bày cụ thể trong phần tiếp theo.

****

**Hình 2 Mô hình xếp tầng các lớp trong giao thức CAN**

## Lớp vật lý

Lớp vật lý sẽ hoàn thiện việc giao tiếp theo các yêu cầu của chuẩn ISO11898-2/5. Để thực hiện việc giao tiếp chẩn đoán trên mạng CAN, tốc độ giao tiếp chẩn đoán cũng chính là tốc độ giao tiếp của ứng dụng và ở đây chọn là 500kbps.

## Lớp liên kết dữ liệu

Lớp liên kết dữ liệu sẽ hoàn thiện việc giao tiếp theo các yêu cầu của chuẩn ISO11898-1. Khối dữ liệu trong CAN trong mỗi gói tin sẽ được cài đặt mặc định chứa 8 bytes và giá trị CAN DLC cũng được gán bằng 8 để tránh việc dàn bit dữ liệu. Đối với những byte dữ liệu không sử dụng của 1 khung dữ liệu CAN sẽ được gán với giá trị 0x55.

Riêng ECU module có thể nhận khung dữ liệu chẩn đoán với độ dài dữ liệu ít hơn 8 byte được gửi từ module Tester.

## Lớp mạng

## Phương pháp định địa chỉ

Mỗi module ECU chỉ hỗ trợ định địa chỉ thông thường và dùng là chế độ địa chỉ 11 bits. Chỉ có chế độ định địa chỉ vật lý được hỗ trợ bởi ECU. Các chế độ địa chỉ chức năng không cần phải được hỗ trợ.

## Cài đặt CAN ID để thực hiện giao tiếp chẩn đoán

* Request ID: 0x712
* Reponse ID; 0x7A2

Module Tester sẽ gửi request thông qua Request ID tới Module ECU. Sau đó, ECU sẽ trả lời yêu cầu bằng Reponse Id.

## Định dạng gói tin chẩn đoán

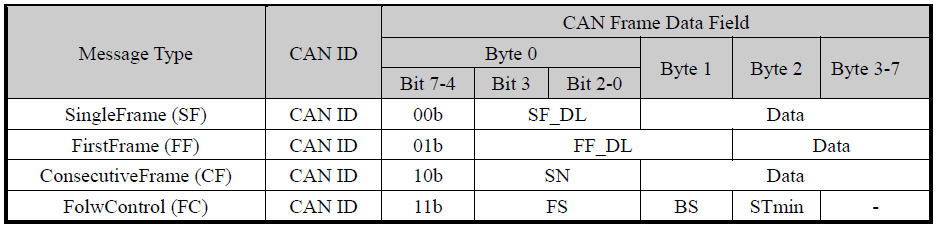
Định dạng khung của gói tin chẩn đoán được mô tả bởi bảng **Bảng 1**. Để tìm hiểu chi tiết hơn, sinh viên có thể tham khảo từ tài liệu CAN\_TP và các tài liệu liên quan khác.

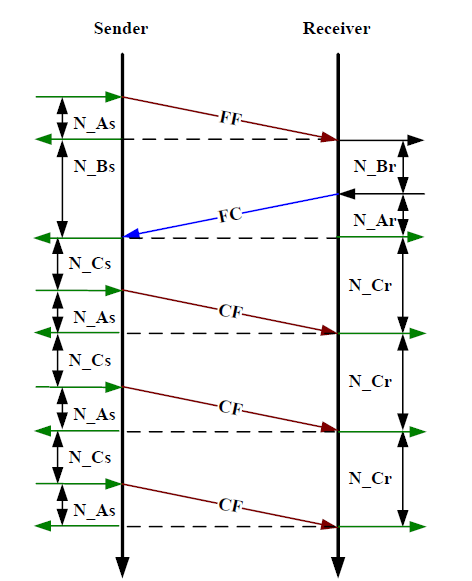
## Định nghĩa các thông số

Những thông số và tiến trình hoạt động của lớp mạng được xác định trong . Để tìm hiểu chi tiết hơn, sinh viên có thể tham khảo từ tài liệu CAN\_TP và các tài liệu liên quan khác.

Khi có nhiều gói tin được yêu cầu, Module gửi và nhận sẽ giao tiếp theo tiến trình như trong **Hình 3**. Giá trị lớn nhất của “FC.Wait Frame Transmissions” (N\_WFTmax) sẽ được gán bằng 0 để khỏi sử dụng FA.Wait.

**Bảng 1 Định dạng gói tin chẩn đoán**

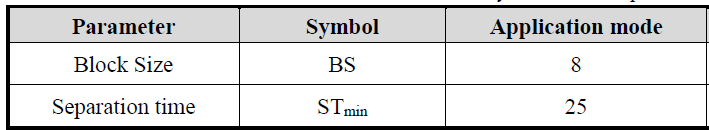




**Hình 3 Tiến trình giao tiếp truyền/nhận các gói tin CAN**

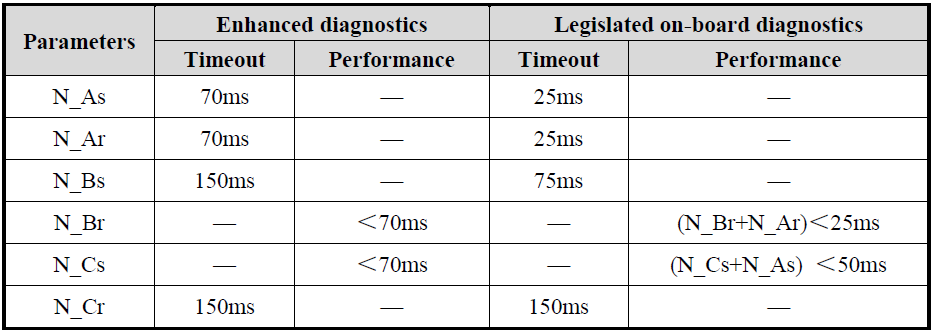
Kích thước block và thời gian giãn cách (ms) được xác định theo :

**Bảng 2 Thông số kích thước khối và thời gian giãn cách**



Trong phạm vi bài thực hành này, sinh viên không cần hiện thực các thông số của lớp mạng. Bảng thông số chủ yếu được dùng để mục đích tham khảo. Tuy nhiên, trong các dự án thực tế, các thông số này sẽ phải được hiện thực.

**Bảng 3 Thông số của lớp mạng**

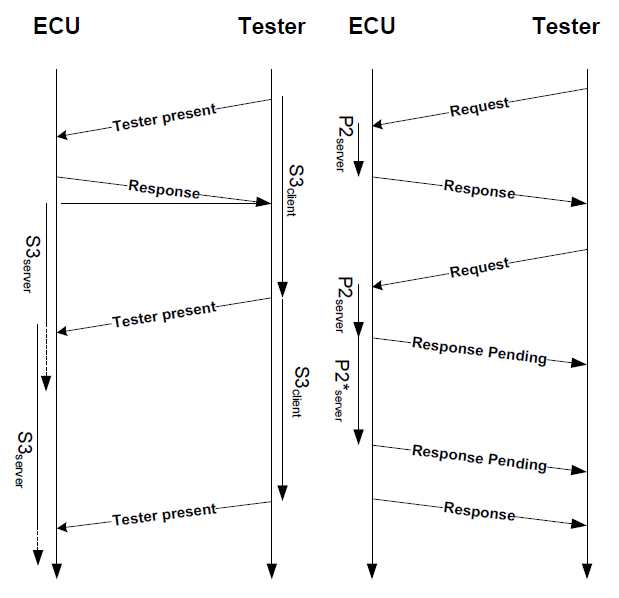


## Lớp chẩn đoán

## Đặc tả chung

## Yêu cầu thời gian

**Hình 4** thể hiện tiến trình giao tiếp theo thời gian giữa Tester và ECU. Tester sẽ gửi yêu cầu và ECU sẽ hồi đáp dựa trên nội dung của yêu cầu.

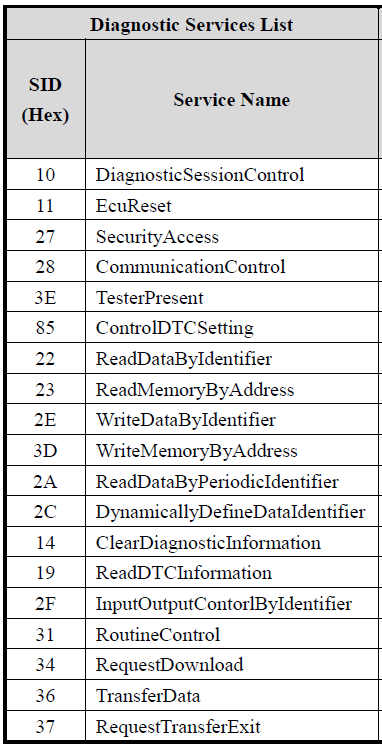


**Hình 4 Tiến trình giao tiếp theo thời gian giữa ECU và Tester**

## Các dịch vụ lớp chẩn đoán

Phần này sẽ xác định những dịch vụ của lớp chẩn đoán cũng như những quy định khi hiện thực lớp này. **Bảng 4** thể hiện tất cả các dịch vụ chẩn đoán và mã định danh (Service Identifier – SID) của các dịch vụ. Trong danh sách, các dịch vụ được sắp xếp theo thứ tự mã định danh được gán cho mỗi dịch vụ.

**Bảng 4 Danh sách các dịch vụ chẩn đoán**



suppressPositveResponseBit (SPRS) thì không được yêu cầu phải hiện thực trong bài thực hành này.

## Negative Response Code (NRC)

Trong bài thực hành này sẽ không yêu cầu sinh viên phải hiện thực NRC. Thay vì hồi đáp bằng NRC, ECU sẽ im lặng và không hồi đáp bất cứ tín hiệu gì.

## Phiên chẩn đoán (Diagnostic session)

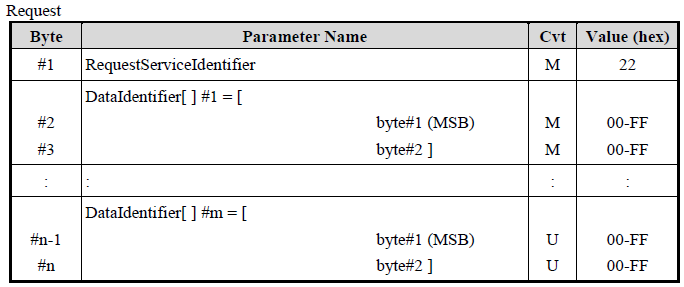
Thông thường, tất cả các ECU sẽ hỗ trợ 3 phiên chẩn đoán. Tuy nhiên, trong phạm vi bài thực hành này, sinh viên không cần hiện thực các phần này.

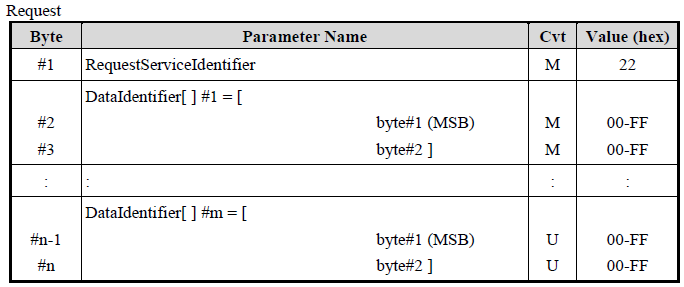
3 phiên chẩn đoán:

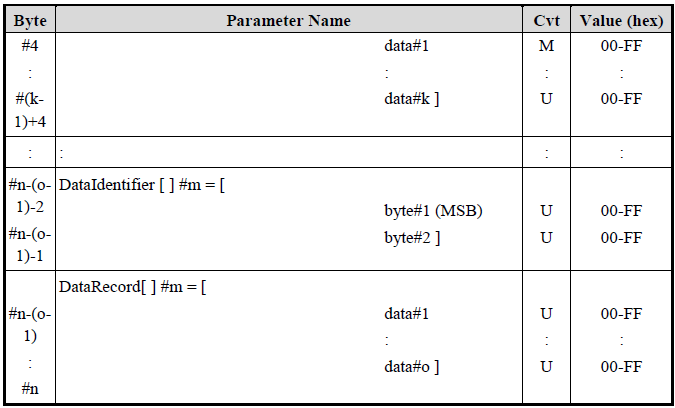
* Phiên mặc định (DefaultSession)
* Phiên lập trình (ProgrammingSession)
* Phiên chẩn đoán mở rộng (ExtendedDiagnosticSession)

## Dịch vụ đọc dữ liệu (ReadDataByIdentifier - 22H)

Gói tin yêu cầu sẽ yêu cầu đọc một khối dữ liệu từ ECU tùy thuộc vào giá trị Data Identifier. ECU sẽ gửi lại khối dữ liệu được yêu cầu thông qua Positive Response Message. Định dạng và định nghĩa của khối dữ liệu (Record Values) sẽ bao gồm các giá trị đầu vào và đầu ra tương tự, các giá trị đầu vào và đầu ra số, các dữ liệu nội bộ và các thông tin trạng thái của hệ thống.





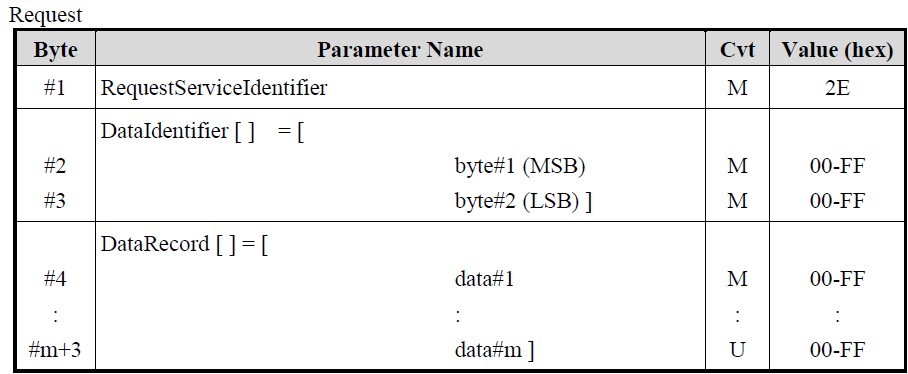


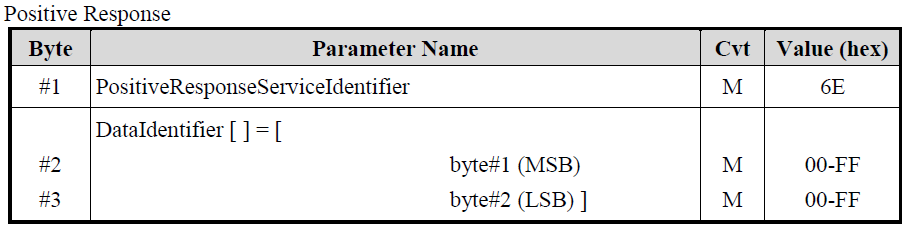
*Lưu ý: Mỗi lần yêu cầu, chỉ một Data Identifier được đọc.*

## Dịch vụ ghi dữ liệu (WriteDataByIdentifier – 2EH)

Dịch vụ ghi dữ liệu cho phép Tester ghi thông tin xuống ECU tại một vị trí bên trong ECU được đặc tả bởi Data Identifier. Những trường hợp có thể của dịch vụ này là:

* Programming configuration information (e.g. VIN).
* Resetting learned values.
* Setting option content.
* Enable or disable function.





*Lưu ý: Mỗi lần yêu cầu, chỉ một Data Identifier được ghi.*

# Bài tập thực hành

1. Trong phần này, sinh viên sẽ hiện thực dịch vụ $22H – Read Data by Identifier để đọc và hiển thị qua UART giá trị ADC:
   1. Tester sẽ gửi yêu cầu tới ECU để đọc giá trị ADC.
   2. ECU sẽ đọc giá trị ADC từ biến trở và gửi lại cho Tester.
   3. Tester nhận được giá trị ADC từ ECU và hiển thị giá trị qua UART.
   4. Mỗi chu kỳ 1 giây, Tester sẽ thực hiện gửi lại yêu cầu đọc ADC 1 lần.
2. Trong phần này, sinh viên sẽ hiện thực dịch vụ $2EH – Write Data by Identifier để ghi một lượng dữ liệu xuống ECU. ECU sẽ hồi đáp và hiển thị các giá trị nhận được lên UART.
   1. Tester sẽ đọc vị trí của Joystick khi Joystick được nhấn.
   2. Sau đó Tester sẽ gửi yêu cầu tới ECU để ghi giá trị dữ liệu tương ứng với vị trí của Joystick. Lưu ý rằng, dữ liệu sẽ không đủ để truyền trong 1 khung gói tin riêng. Sinh viên phải hiện thực quá trình giao tiếp với phương pháp đa gói tin (CAN\_TP).
   3. ECU sẽ nhận yêu cầu và hiển thị toàn bộ gói tin lên UART.
   4. Các quy định về gói tin: Data ID: 0xF002; DID Length: 10 bytes; Format: Binary

|  |  |
| --- | --- |
| Pressed Joystick position | Data (Hex) |
| Middle | 00 00 00 00 00 00 |
| Left | AA AA AA AA AA AA |
| Right | FF FF FF FF FF FF |

# Báo cáo thực hành

Sinh viên viết báo cáo theo mẫu đã được cung cấp để trình bày lại các kết quả đã làm được trong bài thực hành. Giải thích các đoạn chương trình mà sinh viên đã hiện thực để giải quyết các yêu cầu của bài thực hành.

# Tài liệu tham khảo

|  |  |
| --- | --- |
| [1] | "Open405R-C Package A, STM32F4 Development Board," Waveshare Electronics, 2022. [Online]. Available: https://www.waveshare.com/open405r-c-package-a.htm. |
| [2] | STMicroelectronics, [Online]. Available: https://www.st.com/resource/en/datasheet/dm00037051.pdf. [Accessed 2022]. |
| [3] | Waveshare Electronics, [Online]. Available: https://www.waveshare.com/w/upload/f/fe/OpenX05R-C-Schematic.pdf. [Accessed 2022]. |