4.7 CAN Communication

Abstraction

CAN is the most major communication bus among ECUs in-vehicle systems. It is sufficient and reliable to obtain vehicle information such as speed data and HVAC control commands.

Linux has SocketCAN implementation for CAN communication to utilize the popular Berkley socket API. However, on an IVI system, it is expected that a communication function will be provided to meet the following requirements.

- The IVI platform needs to notify CAN data from other ECUs or from external sources to the necessary applications which request to receive the CAN data.
- The IVI platform needs to monitor the periodic reception of CAN data from external sources and to notify the availability of delivery

For these reasons, it is necessary to define a communication function for handling CAN data.

This chapter describes the use cases with CAN Communication function, the functional requirements for realizing the use cases, and the functions of the Basesystem that can be used as a sample implementation.

Use case

In the following table, use cases which need CAN Communication module for services are described.

Table 1

#	Item	Description
UC.CC.1	CAN data send from application to other ECU	When the user presses the button to lower the temperature of the air conditioner, a request is sent to the ECU that controls the temperature of the air conditioner. This causes cold air to come out to lower the interior temperature.
UC.CC.2	CAN data distribution to registered application	IVI system receives the signal data of the steering sensor via CAN bus, when a user is driving the car to park with the gear set reverse checking the predicted trajectory line on the Back-guide monitor.
UC.CC.3	Detect suspension and resume	The CAN communication module receives the CAN data and notifies the registered destination of the data. If the required data is not received for a certain period of time, an unreceived event is sent to the destination, and if the data is received, the destination is notified of the data.

Functional Requirements

This table includes the functional requirements of CAN Communication module.

Table 2

#	Item	Related use cases	Description
RQ.CC.1	Utilization of the function from the application	UC.CC.1	CAN Communication module shall receive the CAN data send request from the application.
RQ.CC.2	CAN data distribution registration	UC.CC.2	CAN Communication module shall be able to register the distribution of applications that want to receive CAN data.
RQ.CC.3	CAN data distribution registration	UC.CC.2	CAN Communication module shall send CAN data to the registered application.
RQ.CC.4	Detect suspension	UC.CC.3	The CAN Communication module shall notify the registered delivery destination of not being received if CAN data is not received within the specified time.
RQ.CC.5	Resume after suspension	UC.CC.3	In the case of RQ.CC.4 status, if the CAN Communication module receives CAN data, it shall notify the registered application of the CAN data.

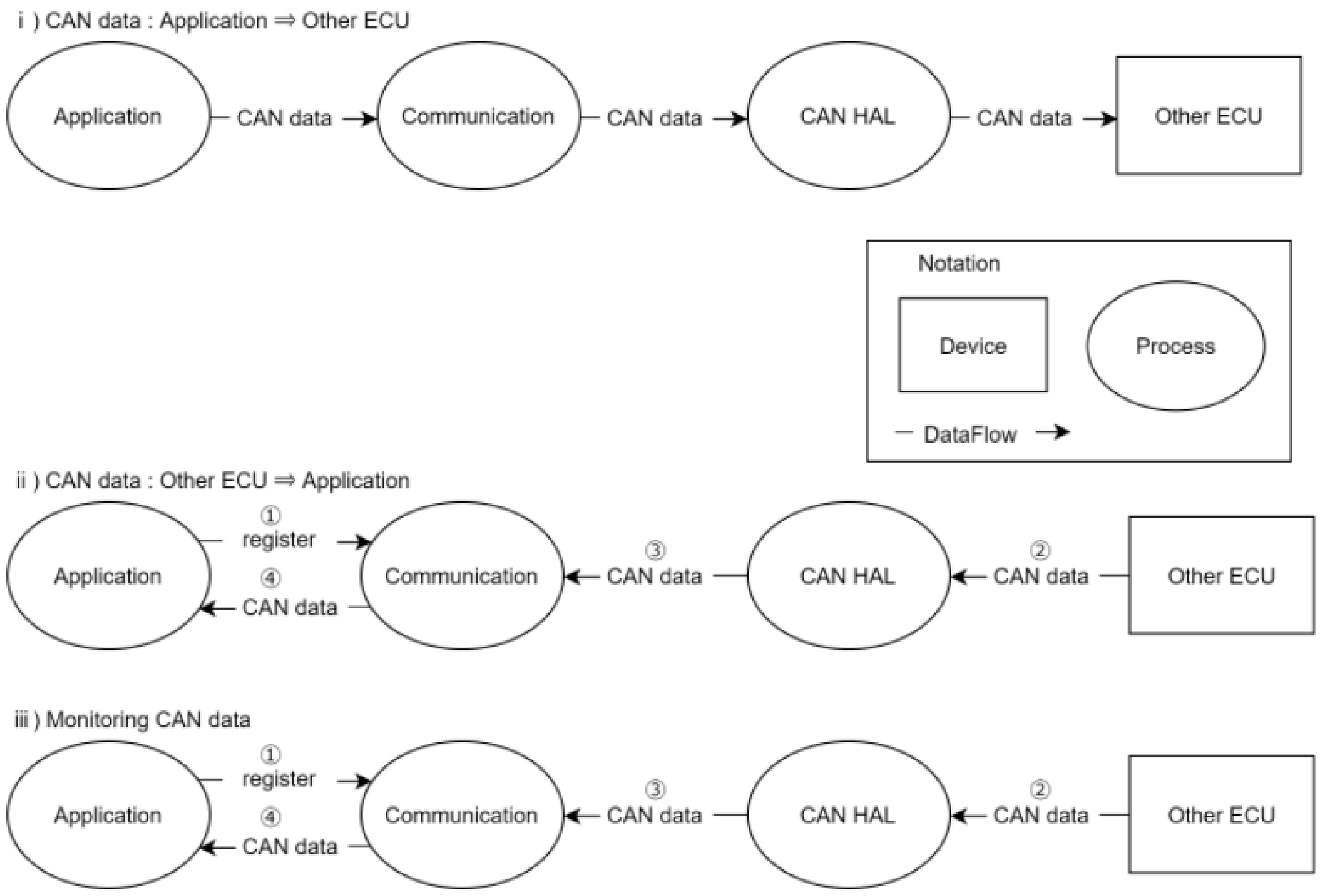
CAN Communication in Basesystem

Reference implementation in Basesystem

In the Basesystem implementation, the functional module that performs CAN communication is Communication. The CAN data flow based on UC.CC.1 and UC.CC.2 of the use case is illustrated in Figure 3. In the Basesystem implementation, the CAN communication functions are implemented via the HAL, which is implemented to allow data exchange independent of the device.

- Send CAN data received from the application to CAN HAL.
- Send CAN data received from CAN HAL to the registered applications.
- Monitor the CAN data communication suspension and notify the registered applications of resume.

Figure 3



The contributed CAN HAL is a Stub with no implementation part, and if you want to use the Communication module, you have to implement the HAL.