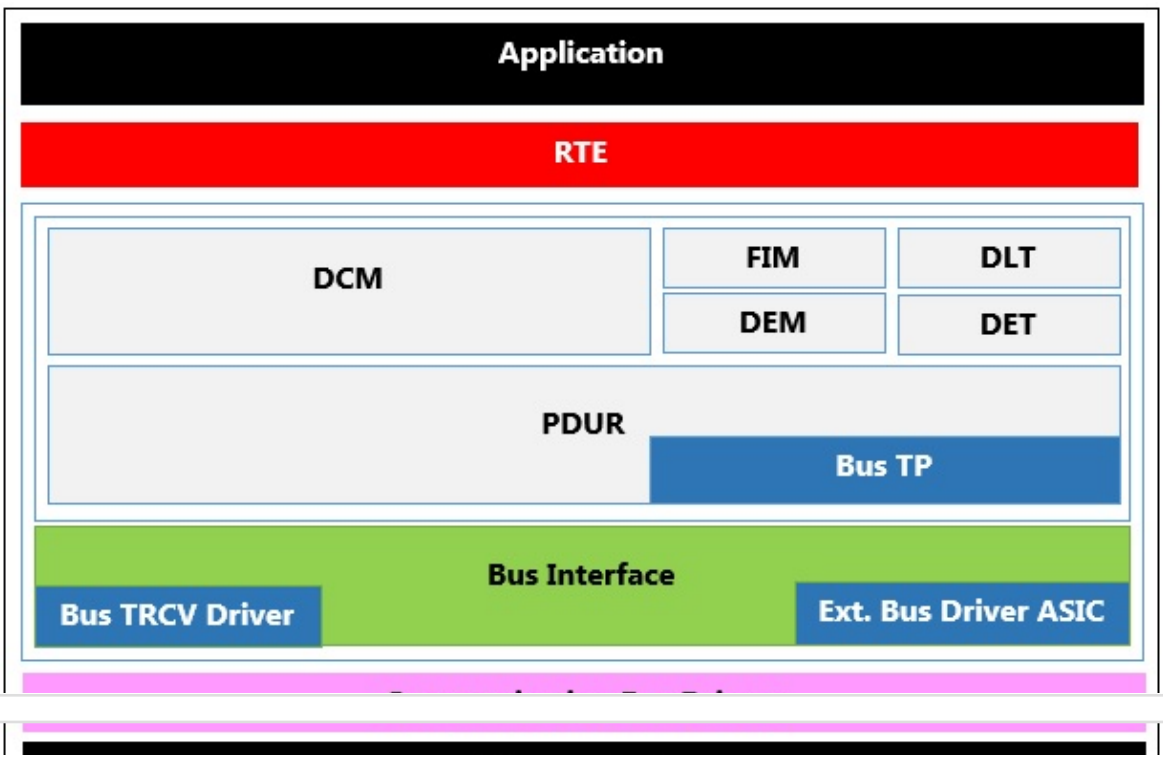
Automotive & Embedded Info

Never Forget Basics Whether its Life or Anything Else ... Basics are Cores. While seeing a Tree how can we forget Seed...

Diagnostic Stack

What is diagnostic?, Its services, session etc., how diagnostic works? All you can see from diagnostic overview and UDS section of this website.

Here scope of this section is to show to the reader the flow of diagnostic stack of AUTOSAR. Below diagram shows AUTOSAR Diagnostic Stack.



Microcontroller

Diagnostic modules brief description:

DCM:

The DCM module ensures diagnostic data flow and manages the diagnostic states, especially diagnostic sessions and security states. Furthermore, the DCM module checks if the diagnostic service request is supported and if the service may be executed in the current session according to the diagnostic states. The DCM module provides the OSI-Layers 5 to 7.

At OSI-level 7, the DCM module provides an extensive set of ISO14229-1 [15] services. In addition, the DCM module provides mechanisms to support the OBD services $01 – $0A. With these services, AUTOSAR OBD functionality is capable of meeting all light duty OBD regulations worldwide (California OBDII, EOBD, Japan OBD, and all others).

At OSI-level 5, the DCM module handles the network-independent sections of the following specifications:

1. ISO15765-3 [18]: Implementation of unified diagnostic services (UDS on CAN).
2. ISO15765-4 [19]: Requirements for emission-related systems, “Session Layer”.

The DCM module is network-independent. All network-specific functionality (the specifics of networks like CAN, LIN, FLEXRAY or MOST) is handled outside of the DCM module. The PDU Router (PDUR) module provides a network-independent interface to the DCM module.

The DCM module receives a diagnostic message from the PDUR module. The DCM module processes and checks internally the diagnostic message. As part of processing the requested diagnostic service, the DCM will interact with other BSW modules or with SW-Components (through the RTE) to obtain requested data or to execute requested commands. This processing is very service-specific. Typically, the DCM will assemble the gathered information and send a message back through the PDUR module.

DCM module as consisting of the following sub modules:

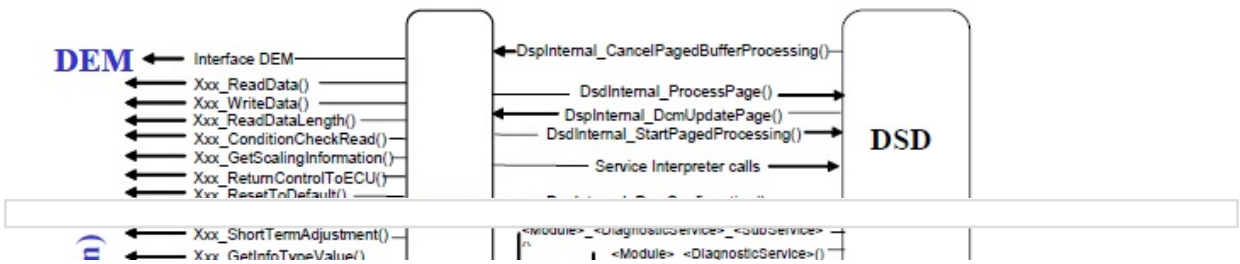
1. Diagnostic Session Layer (DSL) sub module:
   The DSL sub module ensures data flow concerning diagnostic requests and responses, supervises and guarantees diagnostic protocol timing and manages diagnostic states (especially diagnostic session and security).

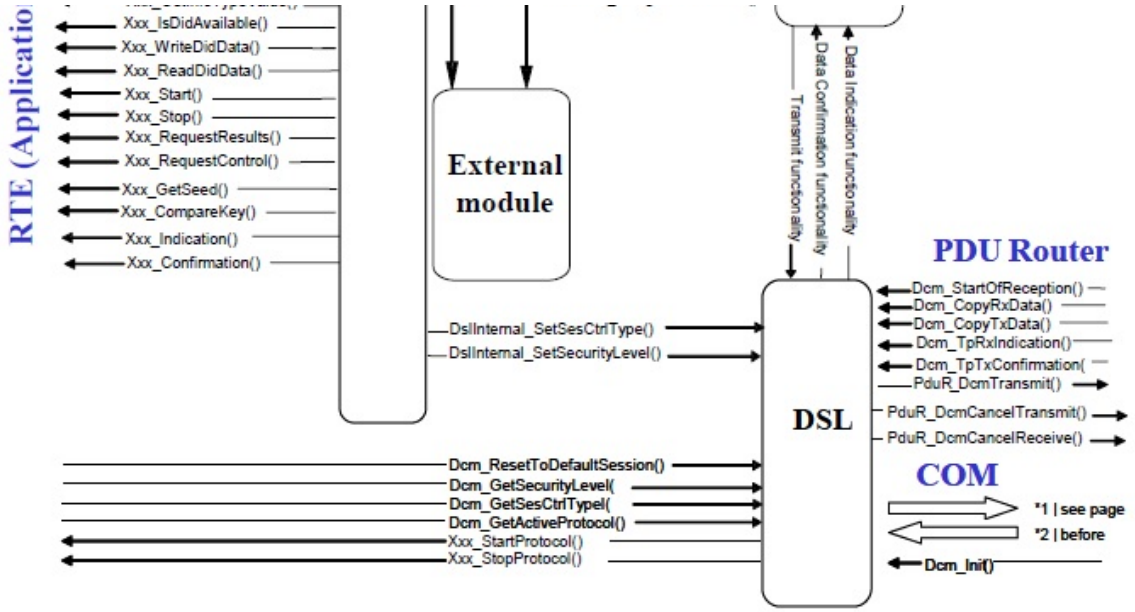
2. Diagnostic Service Dispatcher (DSD) sub module:
   The DSD sub module processes a stream of diagnostic data. The sub module:

   1. Receives a new diagnostic request over a network and forwards it to a data processor.

   2. Transmits a diagnostic response over a network when triggered by the data processor (e.g. by the DSP sub module).

3. Diagnostic Service Processing (DSP) sub module





DCM Sub module Interaction

DEM:

The service component Diagnostic Event Manager (Dem) is responsible for processing and storing diagnostic events (errors) and associated data. Further, the Dem provides fault information to the DCM (e.g. read all stored DTCs from the event memory). The Dem offers interfaces to the application layer and to other BSW modules.

The Diagnostic Event Manager (Dem) handles and stores the events detected by diagnostic monitors in both Software Components (SW-Cs) and Basic software (BSW) modules. The stored event information is available via an interface to other BSW modules or SW-Cs.

DET:

The Default Error Tracer provides functionality to support error detection and tracing of errors during the development and runtime of Software Components and other Basic Software Modules. For this purpose the Default Error Tracer receives and evaluates error messages from these

components and modules.

DLT:

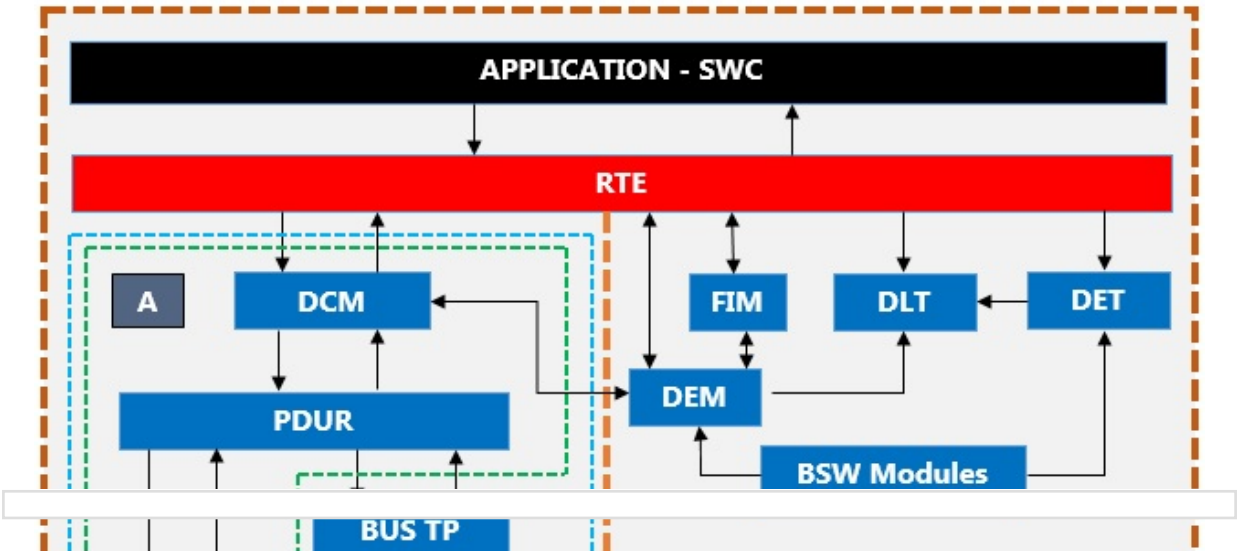
The Diagnostic Log and Trace (DLT) is a Basic Software Module of the Diagnostic Services. It provides a generic Logging and Tracing functionality.

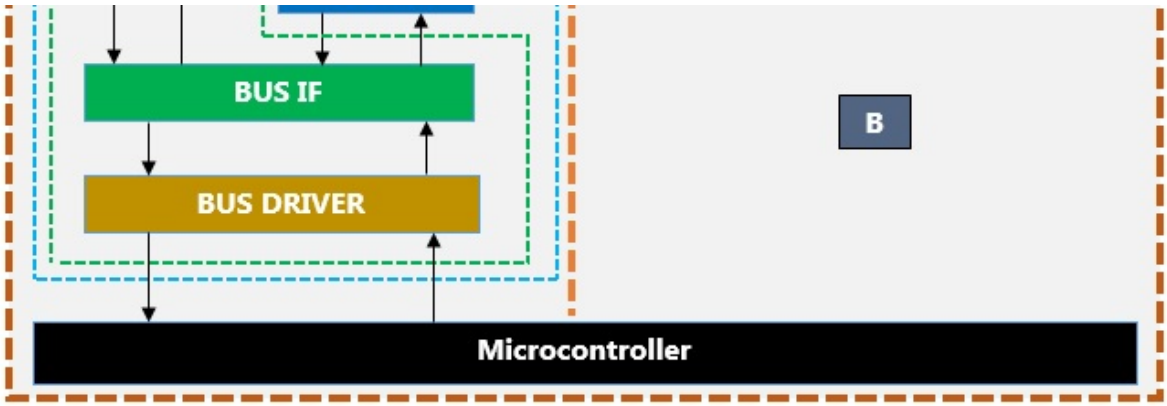
FIM:

The Function Inhibition Manager (FIM) is a Basic Software Modules of the Diagnostic Services. The FiM stands for the evaluation and assignment of events to the required actions for Software Components (e.g. inhibition of specific “Monitors”).

The Dem informs and updates the Function Inhibition Manager (FiM) upon changes of the event status in order to stop or release function entities according to assigned dependencies.

I would like to explain tell flow of communication stack with the below diagram:





DCM: It is in charge of the communication path and execution of diagnostic service resulting in the processing of diagnostic request from an external tester or onboard test system. It forwards requests coming from an external diagnostic scan tool and is further responsible for assembly of response messages (DTC, status information, etc.) which will be transferred to the external diagnostic scan tool afterwards.

DLT: It provides a generic Logging and Tracing functionality.

DET: Development Error Tracer module to which development errors are reported.

DEM: Relevant errors are reported either from Application Layer (resp. SWC) or Basic Software Modules (BSWM).

1. BSWs report the new status of the event with the Dem_ReportErrorStatus API.
2. SWCs report the new status of the event with the Dem_SetEventStatus API (through the RTE)

The Diagnostic Event Manager (Dem) handles and stores the diagnostic event detected by diagnostic monitors in both Software Components(SW-Cs) and Basic Software Modules (BSWM). The stored event information is available via an interface to other BSW modules or SW-Cs.

FIM: The Function Inhibition Manager is responsible for providing a control mechanism for software components and the functionality therein. In this context, a functionality can be built up of the contents of one, several or parts of Runnable Entities with the same set of permission / inhibit conditions. By means of the FIM, the inhibiting of these functionalities can be configured and even modified by calibration. Therefore, the adaptation of a functionality into a new system context with modified physical boundary conditions and influences is significantly enhanced.

The FiM is closely related to the Dem since diagnostic events and their status information are supported as inhibit conditions. Hence, functionality which needs to be stopped in case of a failure, e.g. of a certain sensor, can be represented by a particular identifier. If the failure is detected and the event is reported to the Dem, the FiM then inhibits the FID and therefore the corresponding functionality.

In order to handle the relation of functionality and linked events, the identifier and inhibit conditions of the functionality have been introduced into the SW-C

From above figure and module description we can understand the flow of diagnostic stack.

Share this:

Twitter Facebook

Like

Be the first to like this.

Automotive & Embedded Info / Powered by WordPress.com.