**Basic information on FiM**

**FiM :**

* The Function Inhibition Manager is responsible for providing a control mechanism for software components and also the functionality.
* In this context, functionality can comprise of one, several or parts of runnable entities having same set of permission / inhibit conditions.
* Each functionality is represented by a unique function ID.
* FIM deals with inhibit conditions and provides supporting mechanisms for controlling functionalities within runnables via respective identifiers (FID). Therefore, the FIM and RTE concepts do not interfere with each other.
* The FIM has to deal with the FIDs of the functionalities to provide the automatic checking-mechanism for permission of execution on the demanded sections.

**Different inhibit options in FiM :**

* FIM\_LAST\_FAILED : DEM\_MONITOR\_STATUS\_TF flag of DemMonitorStatus is set
* FIM\_NOT\_TESTED : DEM\_MONITOR\_STATUS\_TNCTOC flag of DemMonitorStatus is set
* FIM\_TESTED : DEM\_MONITOR\_STATUS\_TNCTOC flag of DemMonitorStatus is not set
* FIM\_TESTED\_AND\_FAILED : DEM\_MONITOR\_STATUS\_TF flag of DemMonitorStatus is set and DEM\_MONITOR\_STATUS\_TNCTOC flag is not set

**Function Enable/Disable Conditions :** A functionality or a feature shall be Enabled or Disabled depending on it’s corrsponding FiD flags as below :

* TRUE : FID has permission to run.

FALSE: FID has no permission to run, i.e. shall not be executed.

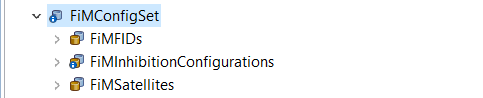
**FiM Configuration**

* Generally we have 2 sub – containers in FiM as shown below :



**FiMConfigSet container :**

* Has following 3 sub – containers :

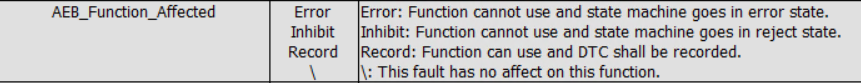


**FiMFIDs container :** This sub container of “**FiMConfigSet container** ” can be used for configuration of different flags corresponding to a feature.

* Generally one flag represents one feature. But it depends upon the requirements from the customer the number of flags to be configured for a functionality.

**Configuration of FiMFIDs in GWM**

In GWM, as per requirements we have multiple flags for a feature as shown in below requirement :



* In mentioned requirement, AEB is an ADAS functionality which shall be affected by a “FAULT” when corresponding fault condition is raised.
* As per the mentioned requirement we shall have 2 flags for AEB as “Error” and “Inhibit”. So there shall be 2 flags configured for ACC as shown below :

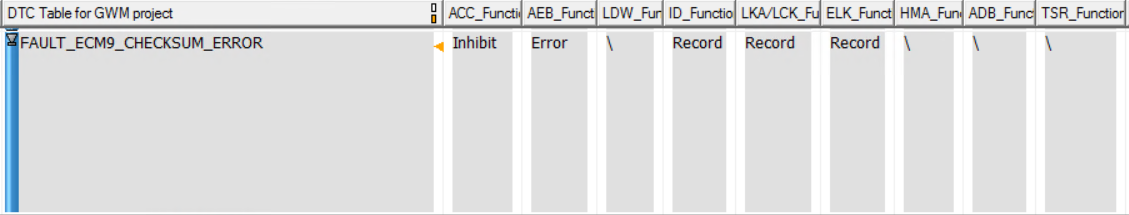


* As shown in above snapshot “AEB\_Error\_Functionality” has FiD index as “4” and “AEB\_Inhibition\_Functionality” has FiD index as “5”.
* The mentioned index for the flags shall be monitored at the time of testing for FiM specifying whether corresponding functionality is Enabled or Disabled.
* As per GWM, there is a difference between a flag configured as Error and Inhibit :
  + Error – when error flag for a functionality is raised, it disables functionality and warns driver that the particular functionality has been disabled.
  + Inhibit – when Inhibit flag for a functionality is raised, it just disables the corresponding functionality.

**Configuration of FiMInhibitionConfigurations in GWM**

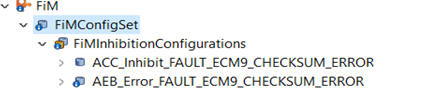
The “FiMInhibitionConfigurations” container shall have all the possible configurations that shall affect an functionality when a particular fault is logged.

* For example we have a requirement as shown below :



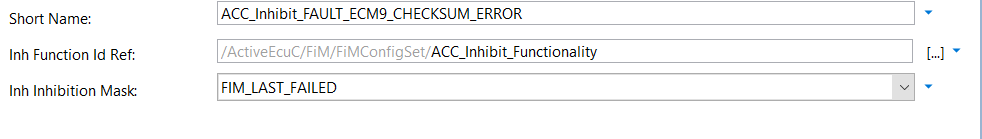
* As per requirement as shown “FAULT\_ECM9\_CHECKSUM\_ERROR” has affect on “ACC as Inhibit” and “AEB as Error”
* So in configuration we shall have 2 flags mapped to “FAULT\_ECM9\_CHECKSUM\_ERROR” in configuration namely “ACC\_Inhibit\_FAULT\_ECM9\_CHECKSUM\_ERROR” and “AEB\_Error\_FAULT\_ECM9\_CHECKSUM\_ERROR”.

As mentioned in the previously slide below is the mentioned configured faults in configurator :



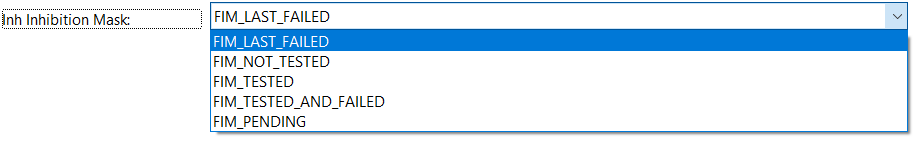
- Configuration of “ACC\_Inhibit\_FAULT\_ECM9\_CHECKSUM\_ERROR” :

- As shown below configuration of a FiM event has 3 parameters as below :

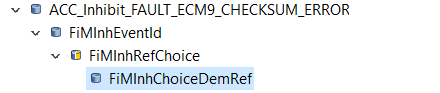


As mentioned in previous slide the 3 parameters shall be configured as for each FiM Event as per requirement :

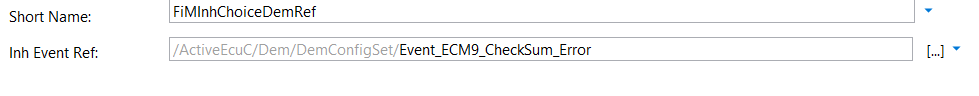
1. **Short Name** : Depends upon the developer but it shall be readable and shall convey the meaning of it’s existence.
2. **Inh Function Id Ref** : is nothing but reference of a flag corresponding to a functionality that shall be affected when this fault is logged as per requirement.
3. **Inh Inhibition Mask :** Itshall depend up on the requirements from the customer. We have multiple options for inhibition mask as shown below



* The description for the above mask is given in the first slide of this presentation.
* As we proceed further to configure our Fault we shall have 2 more sub containers namely “FiMInhEventId” and “FiMInhRefChoice” but the container we have to consider is “FiMInhChoiceDemRef” which is sub container of “FiMInhRefChoice”.



- For the sub container highlighted in above snapshot has following references :

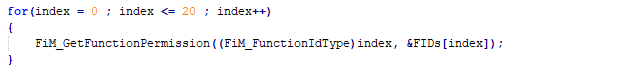


* In the above mentioned snapshot “Inh Event Ref” is nothing but refers to fault “FAULT\_ECM9\_CHECKSUM\_ERROR” and is configured as “Event\_ECM9\_CheckSum\_Error” in DEM.

**Testing of FiM**

* For FiM testing :
  + We need to make sure that the system is Fault free and no faults shall be logged i.e. neither Application Faults nor Core Faults.
  + And a test code as shown below shall be added in order to monitor all the FIDs configured in the Configurator:
  + Test Code Snippet :

* + Test code :
    - uint8 index;
    - uint8 FIDs[21] = {2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2};
* Take an index value and an array of a size equal to configured FIDs as per requirement. In GWM we have 21 FIDs so the array size mentioned is 20 (0..20).
* The test code shall be called periodically monitoring the status of the FIDs. For our testing we used a task that is being called for every 10ms.
* The monitoring of the FIDs shall be done by below mentioned function which is a part of test code which iterates though all the FIDs and updates their status. Find below with the snippet of test code :



Test Code :

for(index = 0 ; index <= 20 ; index++)

{

FiM\_GetFunctionPermission((FiM\_FunctionIdType)index, &FIDs[index]);

}

* FiM\_GetFunctionPermission Function : Prototype is defined as :

Std\_ReturnType FiM\_GetFunctionPermission (FiM\_FunctionIdType FID, boolean\* Permission)

- Function accepts 2 arguments as :

- FID : Function Id as configured in configurator identifies a functionality

Minimum value being 1, if configured as 0: Indicates no functionality

Maximum value may vary as per configuration of FIDs in FIM (Max can be either 255 or 65535)

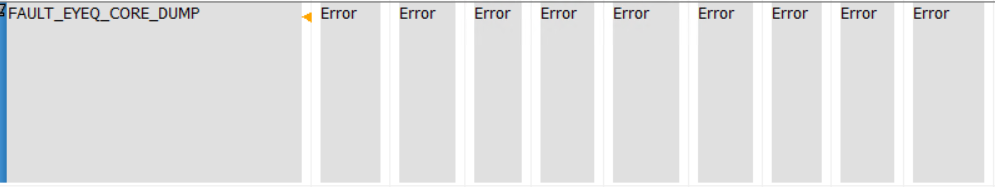
- Permission : This argument notifies whether the corresponding functionality is Enabled or can be accessed.

- TRUE (1) : FID has permission to run

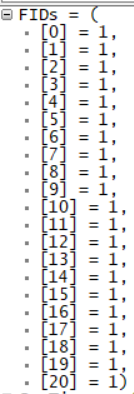
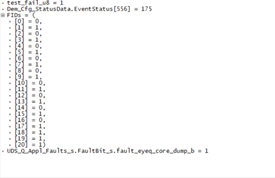
- FALSE (0): FID has no permission to run, i.e. shall not be executed

**Testing of FiM (Example)**

* Let us consider a requirement as below :

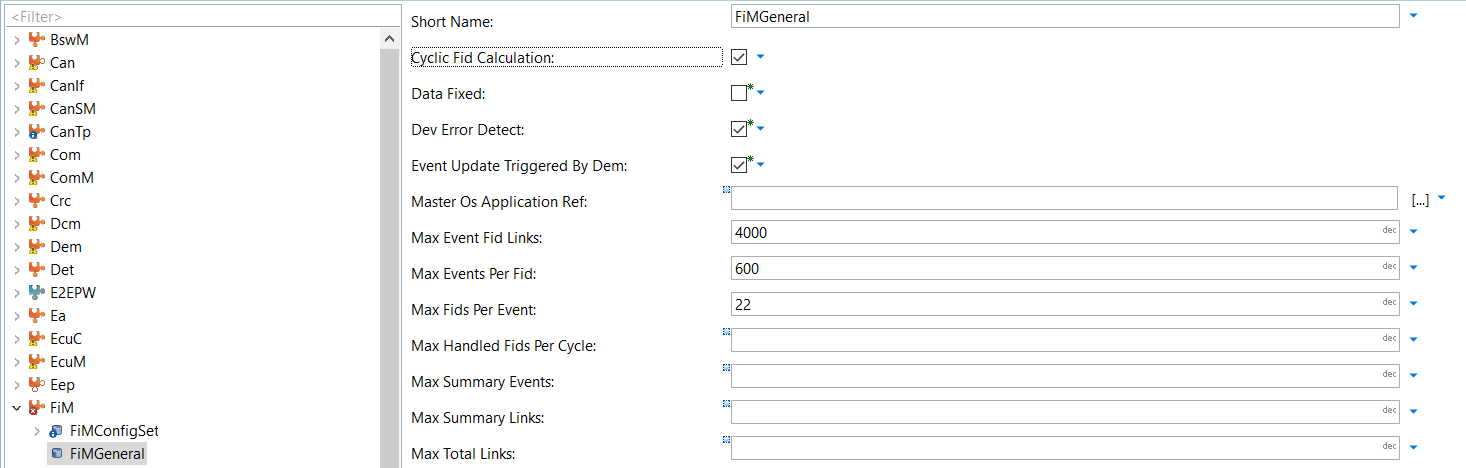


* As per GWM we have 9 flags or Functionality that are supported and the Fault “FAULT\_EYEQ\_CORE\_DUMP” has all the 9 flags as “Errors”.
* This states that when this Fault is set all the 9 flags has to be disabled and hence, no functionality shall be Enabled i.e. all the Error flags must be “0”.
* When there is no fault active in the system all the functionalities are Enabled by default. Hence, we have status of all the FIDs in Trace32 as in snippet (1 and when fault is active the flags get updated as in snippet (2) and the FID reference to the flag that is from configurator shall be seen in (3) :

  Machine generated alternative text:
FiMFlDs 
ACC Error Functionality 
ACC Inhibit Functionality 
ADB Error Functionality 
ADB Inhibit Functionality 
AEB Error Functionality 
AEB Inhibition Functionality 
ELK Error Functionality 
ELK Inhibit Functionality 
HMA Error Functionality 
HMA Inhibit Functionality 
ID Function Error Functionality 
ID Function Inhibit Functionality 
LDW Error Functionality 
LDW Inhibition Functionality 
LKA LCK Error Functionality 
LKA LCK Inhibit Functionality 
TSR Error Functionality 
TSR Inhibit Functionality 
TSR NAV Only Functionality 
TSR VIS Only Functionality 
ACC Functionality One Of Active DTCs 
Function Id 
0 
2 
4 
6 
7 
8 
9 
10 
11 
12 
13 
14 
15 
16 
17 
18 
19 
20 
Satellite Ref 
FiMSatellite 
FiMSatellite 
FiMSatellite 
FiMSatellite 
FiMSatellite 
FiMSatellite 
FiMSatellite 
FiMSatellite 
FiMSatellite 
FiMSatellite 
FiMSatellite 
FiMSatellite 
FiMSatellite 
FiMSatellite 
FiMSatellite 
FiMSatellite 
FiMSatellite 
FiMSatellite 
FiMSatellite 
FiMSatellite 
FiMSatellite 

**Issues faced in FiM**

* One of the Issues which we faced in FiM was related to CPU load being increased drastically.
* This as per analysis was due to a configuration in FiMGeneral wherein we Enabled Cyclic Fid Calculation option.



* In order to reduce the CPU load we disabled this option in configuration.
* And added a partition in ECUC component and configured as shown below.

