TREK SDK API Reference Manual

10/24/2017
SSID.SRP

Copyright © 2017 by Advantech. All Rights Reserved.

Table of contents

2 WARNING 21
3 Introduction 22
3.1 SDK Overview 22
3.1.1 Windows/Linux 22
3.1.2 Android 24
3.2 System Requirement 26
3.2.1 Supported Hardware 26
3.2.2 Supported Platforms 26
3.3 SDK Package Structure 27
3.3.1 Window/Linux 27
3.3.2 Android 28
4 SDK installation 29
4.1 Android 29
4.1.1 Install TREK SDK Services 29
4.1.2 Import TREK SDK Client API 30
5 Application Programming Interface (API) 37
5.1 Library Mapping 37
5.2 SUSI SRV 45
5.2.1 Usage Overview 45
5.2.1.1 Android 45
5.2.2 Structure/Classes 47
5.2.2.1 Android 47
SUSIServiceConnection 47
5.2.3 APIs 48
5.2.3.1 SUSI_IMC_SUSISRV_BindService 48
5.2.3.2 SUSI_IMC_SUSISRV_UnbindService 49
5.2.3.3 SUSI_IMC_SUSISRV_IsServiceInitialized 50
5.2.3.4 SUSI_IMC_SUSISRV_IsServiceConnected 51
5.3 CORE 52
5.3.1 Usage Overview 52
5.3.1.1 Windows/Linux 52

5.3.1.2 Android 53
5.3.2 Structure/Classes 54
5.3.2.1 Windows/Linux 54
IMC_CORE_GETPLATFORMNAME Structure 54
IMC_CORE_ACCESS_BOOTCOUNTER Structure 55
IMC_IMGINFO Structure 56
5.3.2.2 Android 58
IMC_CORE_GETPLATFORMNAME 58
IMC_IMGINFO Structure 59
5.3.3 APIs 60
5.3.3.1 SUSI_IMC_CORE_Init 60
5.3.3.2 SUSI_IMC_CORE_DeInit 61
5.3.3.3 SUSI_IMC_CORE_Available 62
5.3.3.4 SUSI_IMC_CORE_GetLibVersion 63
5.3.3.5 SUSI_IMC_CORE_GetPlatformName 64
5.3.3.6 SUSI_IMC_CORE_AccessBootCounter 65
5.3.3.7 SUSI_IMC_CORE_SoftReset 66
5.3.3.8 SUSI_IMC_CORE_HardReset 67
5.3.3.9 SUSI_IMC_CORE_Reset 68
5.3.3.10 SUSI_IMC_CORE_GetImageInfo 69
5.3.3.11 SUSI_IMC_CORE_RegistrySave 70
5.3.3.12 SUSI_IMC_CORE_RegistryClean 71
5.4 VPM 72
5.4.1 Usage Overview 72
5.4.1.1 Basic Usage 72
Windows/Linux 72
Android 73
5.4.1.2 Get VPM Status 74
Windows/Linux 74
Android 75
5.4.1.3 Ignition Control 76
Windows/Linux 77
Android 78
5.4.1.4 AT Mode / Keep Alive Mode 79
Windows/Linux 80

Android 81
5.4.1.5 Wakeup Source 82
Windows/Linux 82
Android 83
5.4.1.6 Alarm Wakeup 84
Windows/Linux 84
Android 86
5.4.1.7 Low Battery Protection 87
Windows/Linux 88
Android 92
5.4.2 Enumeration 95
5.4.2.1 Windows/Linux 95
ALARM_MODE enum 95
WAKEUP_SOURCE enum 96
SHUTDOWN_SOURCE enum 97
IGNITION_MODE enum 98
5.4.2.2 Android 99
ALARM_MODE enum 99
WAKEUP_SOURCE enum 100
IGNITION_MODE enum 101
CAR_BATTERY_MODE enum 102
5.4.3 Constants 103
5.4.3.1 Android 103
5.4.4 Structure/Classes 104
5.4.4.1 Windows/Linux 104
VPM_RTC_TIME Structure 104
VPM_ALARM_TIME Structure 105
BACKUP_BATTERY_FLAG_INFO Structure 106
PBACKUP_BATTERY_INFO Structure 107
5.4.4.2 Android 108
VPM_RTC_TIME 108
VPM_ALARM_TIME 109
BACKUP_BATTERY_FLAG_INFO 110
BACKUP_BATTERY_INFO 111
BATTERY_INFO 112
5.4.5 APIs 113
5.4.5.1 SUSI_IMC_VPM_Initialize 113

5.4.5.2 SUSI_IMC_VPM_Deinitialize 114
5.4.5.3 SUSI_IMC_VPM_GetFWVersion 115
5.4.5.4 SUSI_IMC_VPM_GetLibVersion 116
5.4.5.5 SUSI_IMC_VPM_GetPlatformName 117
5.4.5.6 SUSI_IMC_VPM_SetIgnitionMode 118
5.4.5.7 SUSI_IMC_VPM_GetIgnitionMode 119
5.4.5.8 SUSI_IMC_VPM_SetIgnOnDelay 120
5.4.5.9 SUSI_IMC_VPM_GetIgnOnDelay 121
5.4.5.10 SUSI_IMC_VPM_SetIgnOffEventDelay 122
5.4.5.11 SUSI_IMC_VPM_GetIgnOffEventDelay 123
5.4.5.12 SUSI_IMC_VPM_SetIgnHardOffDelay 124
5.4.5.13 SUSI_IMC_VPM_GetIgnHardOffDelay 125
5.4.5.14 SUSI_IMC_VPM_SetIgnSuspendHardOffDelay 126
5.4.5.15 SUSI_IMC_VPM_GetIgnSuspendHardOffDelay 127
5.4.5.16 SUSI_IMC_VPM_SetForceShutDown 128
5.4.5.17 SUSI_IMC_VPM_GetForceShutDown 129
5.4.5.18 SUSI_IMC_VPM_SetForceShutDownDelay 130
5.4.5.19 SUSI_IMC_VPM_GetForceShutDownDelay 131
5.4.5.20 SUSI_IMC_VPM_SetPrePowerOffAlarmDelay 132
5.4.5.21 SUSI_IMC_VPM_GetPrePowerOffAlarmDelay 133
5.4.5.22 SUSI_IMC_VPM_SetPrePowerOffAlarm 134
5.4.5.23 SUSI_IMC_VPM_GetPrePowerOffAlarm 135
5.4.5.24 SUSI_IMC_VPM_SetIgnPowerOffEvent 136
5.4.5.25 SUSI_IMC_VPM_SetLBPPowerOffEvent 137
5.4.5.26 SUSI_IMC_VPM_SetLBPLowDelay 138
5.4.5.27 SUSI_IMC_VPM_GetLBPLowDelay 139
5.4.5.28 SUSI_IMC_VPM_SetLBPLowHardDelay 140
5.4.5.29 SUSI_IMC_VPM_GetLBPLowHardDelay 141
5.4.5.30 SUSI_IMC_VPM_SetLBPThresholdFor12VSystem 142
5.4.5.31 SUSI_IMC_VPM_GetLBPThresholdFor12VSystem 143
5.4.5.32 SUSI_IMC_VPM_SetLBPThresholdFor24VSystem 144
5.4.5.33 SUSI_IMC_VPM_GetLBPThresholdFor24VSystem 145
5.4.5.34 SUSI_IMC_VPM_SetPrebootThresholdFor12VSystem 146
5.4.5.35 SUSI_IMC_VPM_GetPrebootThresholdFor12VSystem 147
5.4.5.36 SUSI_IMC_VPM_SetPrebootThresholdFor24VSystem 148

5.4.5.37 SUSI_IMC_VPM_GetPrebootThresholdFor24VSystem 149
5.4.5.38 SUSI_IMC_VPM_GetLBPThresholdInfo 150
5.4.5.39 SUSI_IMC_VPM_GetIgnitionStatus 151
5.4.5.40 SUSI_IMC_VPM_GetCarBatteryVoltage 152
5.4.5.41 SUSI_IMC_VPM_SetCurrentCarBatteryMode 153
5.4.5.42 SUSI_IMC_VPM_GetCurrentCarBatteryMode 154
5.4.5.43 SUSI_IMC_VPM_SetLBPStatus 155
5.4.5.44 SUSI_IMC_VPM_GetLBPStatus 156
5.4.5.45 SUSI_IMC_VPM_SetPrebootLBPStatus 157
5.4.5.46 SUSI_IMC_VPM_GetPrebootLBPStatus 158
5.4.5.47 SUSI_IMC_VPM_SetATMode 159
5.4.5.48 SUSI_IMC_VPM_GetATMode 160
5.4.5.49 SUSI_IMC_VPM_SetKeepAliveMode 161
5.4.5.50 SUSI_IMC_VPM_GetKeepAliveMode 162
5.4.5.51 SUSI_IMC_VPM_SetWakeupSourceControlStatus 163
5.4.5.52 SUSI_IMC_VPM_GetWakeupSourceControlStatus 164
5.4.5.53 SUSI_IMC_VPM_SetShutdownSourceControlStatus 165
5.4.5.54 SUSI_IMC_VPM_GetShutdownSourceControlStatus 166
5.4.5.55 SUSI_IMC_VPM_GetLastWakeupSource 167
5.4.5.56 SUSI_IMC_VPM_SetRTCTime 168
5.4.5.57 SUSI_IMC_VPM_GetRTCTime 169
5.4.5.58 SUSI_IMC_VPM_SetAlarmActive 170
5.4.5.59 SUSI_IMC_VPM_GetAlarmActive 171
5.4.5.60 SUSI_IMC_VPM_SetAlarmTime 172
5.4.5.61 SUSI_IMC_VPM_GetAlarmTime 173
5.4.5.62 SUSI_IMC_VPM_GetBackupBatteryVoltage 174
5.4.5.63 SUSI_IMC_VPM_GetBackupBatteryRemainingCapacity 175
5.4.5.64 SUSI_IMC_VPM_GetBackupBatteryPercentageOfCharge 176
5.4.5.65 SUSI_IMC_VPM_GetBackupBatteryTemperture 177
5.4.5.66 SUSI_IMC_VPM_GetBackupBatteryRemainingTime 178
5.4.5.67 SUSI_IMC_VPM_GetBackupBatteryTimeToFull 179
5.4.5.68 SUSI_IMC_VPM_GetBackupBatteryInfo 180
5.4.5.69 SUSI_IMC_VPM_LoadDefaultValue 181
5.4.5.70 SUSI_IMC_VPM_ForceShutdown 182
5.5 WATCHDOG 183

5.5.1 Usage Overview 183
5.5.1.1 Windows/Linux.....184
5.5.1.2 Android.....185
5.5.2 APIs 186
5.5.2.1 SUSI_IMC_WD_Initialize 186
5.5.2.2 SUSI_IMC_WD_Deinitialize.....187
5.5.2.3 SUSI_IMC_WD_GetLibVersion 188
5.5.2.4 SUSI_IMC_WD_GetRange 189
5.5.2.5 SUSI_IMC_WD_SetTime 190
5.5.2.6 SUSI_IMC_WD_GetTime 191
5.5.2.7 SUSI_IMC_WD_Enabled.....192
5.5.2.8 SUSI_IMC_WD_Trigger.....193
5.6 PERIPHERAL CONTROL.....194
5.6.1 Usage Overview 194
5.6.1.1 Windows/Linux.....194
5.6.1.2 Android.....195
5.6.2 Peripheral Support Table.....196
5.6.3 Enumeration 197
5.6.3.1 Windows/Linux.....197
PERIPHERAL_TYPE enum.....197
SUSI_REAR_VIEW_SRC.....198
5.6.3.2 Android.....198
PERIPHERAL_TYPE enum.....198
SUSI_REAR_VIEW_SRC enum 199
5.6.4 APIs 200
5.6.4.1 SUSI_IMC_PERIPHERALCTRL_Initialize 200
5.6.4.2 SUSI_IMC_PERIPHERALCTRL_Deinitialize 201
5.6.4.3 SUSI_IMC_PERIPHERALCTRL_GetLibVersion 202
5.6.4.4 SUSI_IMC_PERIPHERALCTRL_SetPeripheralControl 203
5.6.4.5 SUSI_IMC_PERIPHERALCTRL_GetPeripheralControl 204
5.6.4.6 SUSI_IMC_PERIPHERALCTRL_SetRearViewSource.....205
5.6.4.7 SUSI_IMC_PERIPHERALCTRL_GetRearViewSource 206
5.6.4.8 SUSI_IMC_PERIPHERALCTRL_SetAutoRearView 207
5.6.4.9 SUSI_IMC_PERIPHERALCTRL_GetAutoRearView.....208

5.7 GSENSOR....209
5.7.1 Usage Overview....209
5.7.1.1 Basic Usage....209
Windows/Linux....209
Android....210
5.7.1.2 G Sensor Alarm Event....211
Windows/Linux....211
Android....213
5.7.1.3 G Sensor Wakeup....214
Windows/Linux....214
Android....216
5.7.2 Enumeration....217
5.7.2.1 Windows/Linux....217
IMC_GSENSOR_RES....217
5.7.2.2 Android....218
IMC_GSENSOR_RES enum....218
5.7.3 Constant....219
5.7.3.1 Android....219
5.7.4 Structure/Classes....219
5.7.4.1 Windows/Linux....219
IMC_GSENSOR_AXIS_INFO Structure....219
5.7.4.2 Android....220
IMC_GSENSOR_DATA....220
5.7.5 APIs....221
5.7.5.1 SUSI_IMC_GSENSOR_Initialize....221
5.7.5.2 SUSI_IMC_GSENSOR_Deinitialize....222
5.7.5.3 SUSI_IMC_GSENSOR_GetLibVersion....223
5.7.5.4 SUSI_IMC_GSENSOR_GetResolution....224
5.7.5.5 SUSI_IMC_GSENSOR_SetResolution....225
5.7.5.6 SUSI_IMC_GSENSOR_GetData....226
5.7.5.7 SUSI_IMC_GSENSOR_GetAlarmThreshold....227
5.7.5.8 SUSI_IMC_GSENSOR_SetAlarmThreshold....228
5.7.5.9 SUSI_IMC_GSENSOR_SetAlarmEvent....229
5.7.5.10 SUSI_IMC_GSENSOR_SetAlarmEventHandler....230

5.7.5.11 SUSI_IMC_GSENSOR_UnsetAlarmEventHandler 231
5.7.5.12 SUSI_IMC_GSENSOR_WaitAlarmEvent 232
5.7.5.13 SUSI_IMC_GSENSOR_AlarmEnabled 233
5.7.5.14 SUSI_IMC_GSENSOR_GetAlarmData 234
5.7.5.15 SUSI_IMC_GSENSOR_GetWakeupThreshold 235
5.7.5.16 SUSI_IMC_GSENSOR_SetWakeupThreshold 236
5.8 DIO 237
5.8.1 Usage Overview 237
5.8.1.1 Windows 237
5.8.1.2 Android 238
5.8.2 Structure/Classes 239
5.8.2.1 Windows/Linux 239
PIN_STATUS 239
5.8.2.2 Android 240
PIN_STATUS 240
5.8.3 APIs 241
5.8.3.1 SUSI_IMC_DIO_Initialize 241
5.8.3.2 SUSI_IMC_DIO_Deinitialize 242
5.8.3.3 SUSI_IMC_DIO_GetLibVersion 243
5.8.3.4 SUSI_IMC_DIO_ReadDIPin 244
5.8.3.5 SUSI_IMC_DIO_WriteDOPin 245
5.8.3.6 SUSI_IMC_DIO_GetVoiceSourceControlPin 246
5.8.3.7 SUSI_IMC_DIO_SetVoiceSourceControlPin 247
5.8.3.8 SUSI_IMC_DIO_GetBluetoothEnablePin 248
5.8.3.9 SUSI_IMC_DIO_SetBluetoothEnablePin 249
5.8.3.10 SUSI_IMC_DIO_GetVoiceModuleSelectPin 250
5.8.3.11 SUSI_IMC_DIO_SetVoiceModuleSelectPin 251
5.9 IIC 252
5.9.1 Usage Overview 252
5.9.1.1 Windows/Linux 252
5.9.2 Structure/Classes 253
5.9.2.1 Windows/Linux 253
IMC_IIC_READ Structure 253
IMC_IIC_WRITE Structure 254

IMC_IIC_WRITE_READ_COMBINE Structure 255
5.9.3 APIs 256
5.9.3.1 SUSI_IMC_IIC_GetLibVersion 256
5.9.3.2 SUSI_IMC_IIC_Available 257
5.9.3.3 SUSI_IMC_IIC_Read 258
5.9.3.4 SUSI_IMC_IIC_Write 259
5.9.3.5 SUSI_IMC_IIC_WriteReadCombine 260
5.10 EEPROM 261
5.10.1 Usage Overview 261
5.10.1.1 Windows/Linux 261
5.10.1.2 Android 263
5.10.2 APIs 264
5.10.2.1 SUSI_IMC_EEPROM_Initialize 264
5.10.2.2 SUSI_IMC_EEPROM_Deinitialize 265
5.10.2.3 SUSI_IMC_EEPROM_GetLibVersion 266
5.10.2.4 SUSI_IMC_EEPROM_GetEEPROMSize 267
5.10.2.5 SUSI_IMC_EEPROM_ReadByte 268
5.10.2.6 SUSI_IMC_EEPROM_WriteByte 269
5.10.2.7 SUSI_IMC_EEPROM_ReadMultiByte 270
5.10.2.8 SUSI_IMC_EEPROM_WriteMultiByte 272
5.11 VCIL 274
5.11.1 Usage Overview 274
5.11.1.1 Windows/ Linux/ Android 275
5.11.2 Enumeration 276
5.11.2.1 Windows/Linux 276
VCIL_CAN_CHANNEL_MODE enum 276
VCIL_J1708_CHANNEL_MODE enum 277
5.11.2.2 Android 278
VCIL_CAN_CHANNEL_MODE enum 278
VCIL_J1708_CHANNEL_MODE enum 279
5.11.3 Constant 280
5.11.3.1 Android 280
5.11.4 APIs 280
5.11.4.1 SUSI_IMC_VCIL_Initialize 280

5.11.4.2 SUSI_IMC_VCIL_InitializeEx 282
5.11.4.3 SUSI_IMC_VCIL_Deinitialize 283
5.11.4.4 SUSI_IMC_VCIL_GetLibVersion 284
5.11.4.5 SUSI_IMC_VCIL_GetFWVersion 285
5.11.4.6 SUSI_IMC_VCIL_ResetModule 286
5.11.4.7 SUSI_IMC_VCIL_ModuleControl 287
5.12 CAN 289
5.12.1 Usage Overview 289
5.12.1.1 Basic Usage 290
Windows/Linux 290
Android 290
5.12.1.2 CAN Bus Speed Setting 291
Windows/ Linux/ Android 291
5.12.1.3 CAN Message Reading 292
Windows/Linux 292
Android 294
5.12.1.4 CAN Message Writing 296
Windows/ Linux/ Android 296
5.12.1.5 CAN Acceptance filter Settings 297
Windows/ Linux/ Android 301
5.12.2 Enumeration 303
5.12.2.1 Windows/Linux 303
CAN_MESSAGE_TYPE enum 303
CAN_SPEED enum 304
CAN_BUS_MODE enum 305
5.12.2.2 Android 306
CAN_MESSAGE_TYPE enum 306
CAN_SPEED enum 307
CAN_BUS_MODE enum 308
5.12.3 Structure/Class 309
5.12.3.1 Windows/Linux 309
IMC_CAN_MSG_OBJECT Structure 309
IMC_CAN_MASK_OBJECT Structure 310
IMC_CAN_ERROR_STATUS_OBJECT Structure 311
5.12.3.2 Android 312

IMC_CAN_MSG_OBJECT....312
IMC_CAN_MASK_OBJECT....313
IMC_CAN_ERROR_STATUS_OBJECT....314
5.12.4 APIs....315
5.12.4.1 SUSI_IMC_CAN_Read....315
5.12.4.2 SUSI_IMC_CAN_Write....316
5.12.4.3 SUSI_IMC_CAN_GetBusErrorStatus....317
5.12.4.4 SUSI_IMC_CAN_SetEvent....318
5.12.4.5 SUSI_IMC_CAN_SetEventHandler....319
5.12.4.6 SUSI_IMC_CAN_UnsetEventHandler....320
5.12.4.7 SUSI_IMC_CAN_WaitEvent....321
5.12.4.8 SUSI_IMC_CAN_SetBitTiming....322
5.12.4.9 SUSI_IMC_CAN_SetBitTimingSilence....324
5.12.4.10 SUSI_IMC_CAN_GetBitTiming....325
5.12.4.11 SUSI_IMC_CAN_SetMessageMask....327
5.12.4.12 SUSI_IMC_CAN_GetMessageMask....328
5.12.4.13 SUSI_IMC_CAN_RemoveMessageMask....329
5.12.4.14 SUSI_IMC_CAN_ResetMessageMask....330
5.13 J1939....331
5.13.1 Usage Overview....331
5.13.1.1 Basic Usage....331
Windows/Linux....331
Android....331
5.13.1.2 J1939 Transmit Config Settings....332
Windows/ Linux/ Android....332
5.13.1.3 CAN Bus Speed Setting....333
5.13.1.4 J1939 Message Reading....334
Windows/Linux....334
Android....336
5.13.1.5 J1939 Message Writing....338
Windows/ Linux/ Android....338
5.13.1.6 J1939 Acceptance filter Settings....339
Windows/ Linux/ Android....340
5.13.2 Constant....342

5.13.2.1 Android 342
5.13.3 Structure/Class 342
5.13.3.1 Windows/Linux 342
IMC_J1939_MSG_OBJECT Structure 342
IMC_J1939_TRANSMIT_CONFIG Structure 344
5.13.3.2 Android 345
IMC_J1939_MSG_OBJECT 345
IMC_J1939_TRANSMIT_CONFIG 346
5.13.4 APIs 347
5.13.4.1 SUSI_IMC_J1939_Read 347
5.13.4.2 SUSI_IMC_J1939_Write 348
5.13.4.3 SUSI_IMC_J1939_SetEvent 349
5.13.4.4 SUSI_IMC_J1939_SetEventHandler 350
5.13.4.5 SUSI_IMC_J1939_UnsetEventHandler 351
5.13.4.6 SUSI_IMC_J1939_WaitEvent 352
5.13.4.7 SUSI_IMC_J1939_AddMessageFilter 353
5.13.4.8 SUSI_IMC_J1939_GetMessageFilter 354
5.13.4.9 SUSI_IMC_J1939_RemoveMessageFilter 356
5.13.4.10 SUSI_IMC_J1939_RemoveAllMessageFilter 357
5.13.4.11 SUSI_IMC_J1939_SetTransmitConfig 358
5.13.4.12 SUSI_IMC_J1939_GetTransmitConfig 359
5.14 OBD2 360
5.14.1 Usage Overview 360
5.14.1.1 Basic Usage 360
Windows/Linux 360
Android 360
5.14.1.2 CAN Bus Speed Setting 361
5.14.1.3 OBD2 Message Reading 362
Windows/Linux 362
Android 364
5.14.1.4 OBD2 Message Writing 366
Windows/ Linux/ Android 366
5.14.1.5 OBD2 Acceptance filter Settings 367
Windows/ Linux/ Android 368

5.14.2 Constant 370
5.14.2.1 Android 370
5.14.3 Structure/Class 371
5.14.3.1 Windows/Linux 371
IMC_OBD2_MSG_OBJECT Structure(deprecated) 371
IMC_OBD2_MSG_EX_OBJECT Structure 372
5.14.3.2 Android 373
IMC_OBD2_MSG_OBJECT 373
5.14.4 APIs 374
5.14.4.1 SUSI_IMC_OBD2_Read(deprecated) 374
5.14.4.2 SUSI_IMC_OBD2_ReadEx 375
5.14.4.3 SUSI_IMC_OBD2_Write((deprecated) 376
5.14.4.4 SUSI_IMC_OBD2_WriteEx 377
5.14.4.5 SUSI_IMC_OBD2_SetEvent 378
5.14.4.6 SUSI_IMC_OBD2_SetEventHandler 379
5.14.4.7 SUSI_IMC_OBD2_UnsetEventHandler 380
5.14.4.8 SUSI_IMC_OBD2_WaitEvent 381
5.14.4.9 SUSI_IMC_OBD2_AddMessageFilter 382
5.14.4.10 SUSI_IMC_OBD2_GetMessageFilter 383
5.14.4.11 SUSI_IMC_OBD2_RemoveMessageFilter 385
5.14.4.12 SUSI_IMC_OBD2_RemoveAllMessageFilter 386
5.15 J1708 387
5.15.1 Usage Overview 387
5.15.1.1 Basic Usage 387
Windows/Linux 387
Android 387
5.15.1.2 J1708 Message Reading 388
Windows/Linux 388
Android 390
5.15.1.3 J1708 Message Writing 392
Windows/ Linux/ Android 392
5.15.1.4 J1708 Acceptance filter Settings 393
Windows/ Linux/ Android 394
5.15.2 Constant 396

5.15.2.1 Android 396
5.15.3 Structure/Class 397
5.15.3.1 Windows/Linux 397
IMC_J1708_MSG_OBJECT Structure 397
5.15.3.2 Android 398
IMC_J1708_MSG_OBJECT 398
5.15.4 APIs 399
5.15.4.1 SUSI_IMC_J1708_Read 399
5.15.4.2 SUSI_IMC_J1708_Write 400
5.15.4.3 SUSI_IMC_J1708_SetEvent 401
5.15.4.4 SUSI_IMC_J1708_SetEventHandler 402
5.15.4.5 SUSI_IMC_J1708_UnsetEventHandler 403
5.15.4.6 SUSI_IMC_J1708_WaitEvent 404
5.15.4.7 SUSI_IMC_J1708_AddMessageFilter 405
5.15.4.8 SUSI_IMC_J1708_GetMessageFilter 406
5.15.4.9 SUSI_IMC_J1708_RemoveMessageFilter 407
5.15.4.10 SUSI_IMC_J1708_RemoveAllMessageFilter 408
5.16 J1587 409
5.16.1 Usage Overview 409
5.16.1.1 Basic Usage 409
Windows/Linux 409
Android 409
5.16.1.2 J1587 Message Reading 410
Windows/Linux 410
Android 412
5.16.1.3 J1587 Message Writing 414
Windows/ Linux/ Android 414
5.16.1.4 J1587 Acceptance filter Settings 415
Windows/ Linux/ Android 416
5.16.2 Constant 418
5.16.2.1 Android 418
5.16.3 Structure/Class 419
5.16.3.1 Windows/Linux 419

IMC_J1587_MSG_OBJECT Structure....419
5.16.3.2 Android....420
IMC_J1587_MSG_OBJECT....420
5.16.4 APIs....421
5.16.4.1 SUSI_IMC_J1587_Read....421
5.16.4.2 SUSI_IMC_J1587_Write....422
5.16.4.3 SUSI_IMC_J1587_SetEvent....423
5.16.4.4 SUSI_IMC_J1587_SetEventHandler....424
5.16.4.5 SUSI_IMC_J1587_UnsetEventHandler....425
5.16.4.6 SUSI_IMC_J1587_WaitEvent....426
5.16.4.7 SUSI_IMC_J1587_AddMessageFilter....427
5.16.4.8 SUSI_IMC_J1587_GetMessageFilter....428
5.16.4.9 SUSI_IMC_J1587_RemoveMessageFilter....429
5.16.4.10 SUSI_IMC_J1587_RemoveAllMessageFilter....430
5.17 SDP SRV....431
5.17.1 Usage Overview....431
5.17.1.1 Android....431
5.17.2 Constant....433
5.17.2.1 Android....433
5.17.3 Structure/Classes....434
5.17.3.1 Android....434
SDPServiceConnection....434
5.17.4 APIs....435
5.17.4.1 SUSI_IMC_SDPSRV_BindService....435
5.17.4.2 SUSI_IMC_SDPSRV_UnbindService....436
5.17.4.3 SUSI_IMC_SDPSRV_IsServiceInitialized....437
5.17.4.4 SUSI_IMC_SDPSRV_IsServiceConnected....438
5.18 CONTROL PANEL....439
5.18.1 Usage Overview....439
5.18.1.1 Windows/Linux....439
5.18.1.2 Android....440
5.18.2 Enumeration....441
5.18.2.1 Android....441

CONTROL_PANEL_RESET_MODE enum....441
5.18.3 Structure/Classes....442
5.18.3.1 Windows/Linux....442
IMC_CONTROLPANEL_FIRMWARE_INFO Structure....442
5.18.3.2 Android....443
IMC_CONTROLPANEL_FIRMWARE_INFO....443
5.18.4 APIs....444
5.18.4.1 SUSI_IMC_CONTROLPANEL_GetVersion....444
5.18.4.2 SUSI_IMC_CONTROLPANEL_Initialize....445
5.18.4.3 SUSI_IMC_CONTROLPANEL_Deinitialize....446
5.18.4.4 SUSI_IMC_CONTROLPANEL_GetFirmwareInformation....447
5.18.4.5 SUSI_IMC_CONTROLPANEL_ResetFirmware....448
5.18.4.6 SUSI_IMC_CONTROLPANEL_LoadDefaultValue....449
5.18.4.7 SUSI_IMC_CONTROLPANEL_TEMPERATURE_GetValue....450
5.19 LIGHT SENSOR API FUNCTIONS....451
5.19.1 Usage Overview....451
5.19.1.1 Basic Usage....451
Windows/Linux....451
Android....451
5.19.1.2 Light sensor status....452
Windows/Linux....452
Android....453
5.19.2 APIs....454
5.19.2.1 SUSI_IMC_LIGHTSENSOR_GetLibVersion....454
5.19.2.2 SUSI_IMC_LIGHTSENSOR_GetStatus....455
5.20 TEMPERATURE SENSOR API FUNCTIONS....456
5.20.1 Usage Overview....456
5.20.1.1 Windows/Linux....456
5.20.2 APIs....457
5.20.2.1 SUSI_IMC_TEMPERATURESENSOR_Initialize....457
5.20.2.2 SUSI_IMC_TEMPERATURESENSOR_Deinitialize....458
5.20.2.3 SUSI_IMC_TEMPERATURESENSOR_GetLibVersion....459
5.20.2.4 SUSI_IMC_TEMPERATURESENSOR_GetCPUCore1Temperature....460

5.20.2.5 SUSI_IMC_TEMPERATURESENSOR_GetCPUCore2Temperature.....461
5.20.2.6 SUSI_IMC_TEMPERATURESENSOR_GetSystem1Temperature.....462
5.21 BRIGHTNESS API FUNCTIONS.....463
5.21.1 Usage Overview 463
5.21.1.1 Basic Usage.....463
Windows/Linux.....463
Android.....463
5.21.1.2 Brightness Setting 464
Windows/Linux.....464
Android.....466
5.21.2 Structure/Classes.....467
5.21.2.1 Windows/Linux.....467
IMC_BRIGHTNESS_LEVEL Structure.....467
5.21.2.2 Android.....468
IMC_BRIGHTNESS_LEVEL 468
5.21.3 APIs.....469
5.21.3.1 SUSI_IMC_BRIGHTNESS_SetDutyCycle.....469
5.21.3.2 SUSI_IMC_BRIGHTNESS_GetDutyCycle.....470
5.21.3.3 SUSI_IMC_BRIGHTNESS_SetLevel.....471
5.21.3.4 SUSI_IMC_BRIGHTNESS_GetLevel.....473
5.22 HOTKEY API FUNCTIONS.....474
5.22.1 Usage Overview 474
5.22.1.1 Basic Usage.....474
Windows/Linux.....474
Android.....474
5.22.1.2 Hotkey Status Getting 475
Windows/Linux.....475
Android.....477
5.22.1.3 Hotkey LED Duty Cycle Setting 480
Windows/Linux.....480
Android.....481
5.22.2 Structure/Classes.....482
5.22.2.1 Windows/Linux.....482
IMC_HOTKEY_MSG_OBJECT Structure.....482

5.22.2.2 Android 483
IMC_HOTKEY_MSG_OBJECT 483
5.22.3 APIs....484
5.22.3.1 SUSI_IMC_HOTKEY_Read 484
5.22.3.2 SUSI_IMC_HOTKEY_RegisterOnPressAndReleaseEventMonitor (Windows/Linux) 485
5.22.3.3 SUSI_IMC_HOTKEY_ClearOnPressEventMonitor (Windows/Linux)....486
5.22.3.4 SUSI_IMC_HOTKEY_RegisterOnPressAndReleaseEventMonitor (Android) 487
5.22.3.5 SUSI_IMC_HOTKEY_ClearOnPressEventMonitor (Android)....488
5.22.3.6 SUSI_IMC_HOTKEY_EnableEventMonitor 489
5.22.3.7 SUSI_IMC_HOTKEY_DisableEventMonitor 490
5.22.3.8 SUSI_IMC_HOTKEY_SetLedDutyCycle....491
5.22.3.9 SUSI_IMC_HOTKEY_GetLedDutyCycle 492
5.23 DISPLAY 493
5.23.1 Usage Overview 493
5.23.1.1 Windows/Linux....493
5.23.2 Structure/Classes....495
5.23.2.1 Windows/Linux....495
IMC_DISPLAY_GET_BRIGHT_RANGE Structure....495
5.23.3 APIs....496
5.23.3.1 SUSI_IMC_DISPLAY_GetLibVersion....496
5.23.3.2 SUSI_IMC_DISPLAY_Available....497
5.23.3.3 SUSI_IMC_DISPLAY_ScreenOn....498
5.23.3.4 SUSI_IMC_DISPLAY_ScreenOff....499
5.23.3.5 SUSI_IMC_DISPLAY_GetBrightRange....500
5.23.3.6 SUSI_IMC_DISPLAY_GetBright 501
5.23.3.7 SUSI_IMC_DISPLAY_SetBright....502
6 RETURN CODE LIST 503
6.1 GENERAL ERROR CODES 503
6.2 CAN API ERROR CODE....504
6.3 VPM API ERROR CODE 505
6.4 G-SENSOR API ERROR CODE 505
6.5 DISPLAY API ERROR CODE 506

6.6 IIC API ERROR CODE 506

2 WARNING

Information in this document is subject to change without prior notice.
No part of this publication may be reproduced, transmitted, stored in a retrieval system, nor translated into any human or computer language, in any form or by any means, electronic, mechanical, magnetic, optical, chemical, manual, or otherwise, without the prior written permission of Advantech Co., Ltd.
All trademarks mentioned are proprietary of their respective owners.

3 Introduction

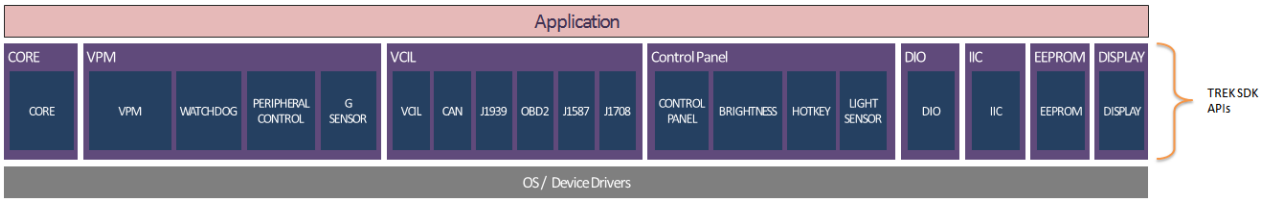
The TREK SDK (Software Development Kit) is a set of software libraries that support features such as Controller Area Network (CAN), J1939, and VPM etc., helps you to make applications for various purpose on TREK devices.

3.1 SDK Overview

The TREK SDK is a set of software libraries which provides APIs for controlling various functions of the TREK device.

3.1.1 Windows/Linux

The following figure describes the software stack of TREK SDK:



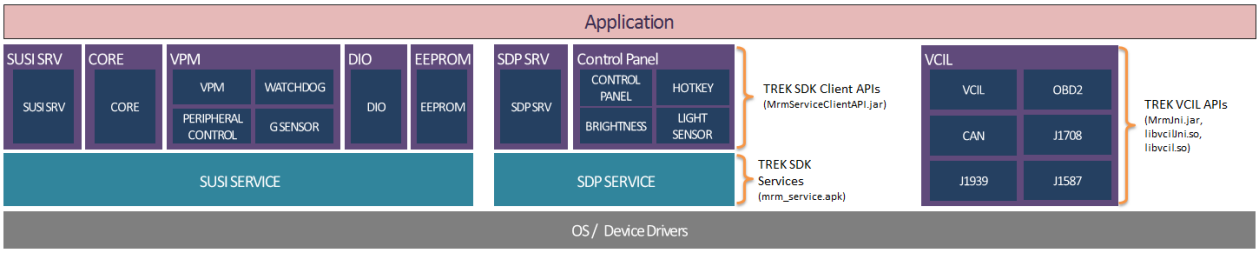
TREK SDK provides modules of APIs for you to control TREK device:

- **CORE**
  - CORE APIs - SDK core functions. Reset device, get status and configure system registry.
- **VPM (Vehicle Power Management)**
  - VPM APIs - Vehicle Power Management(VPM) related functions such as boot control, Ignition control, event delay adjustments, low battery protection and ETC..
  - WATCHDOG APIs - Simple APIs for you to construct your hardware revive mechanism.
  - PERIPHERAL CONTROL APIs - Control/configure peripheral devices.
  - G-SENSOR APIs - Read data from Gravity sensor.
- **VCIL (Vehicle Communication Interface Layer)**
  - VCIL APIs - Configuration physical channel for specific protocol.
  - CAN APIs - Read / write data with CAN protocol.
  - J1939 APIs - Read / write data with J1939 protocol.
  - OBD2 APIs - Read / write data with OBD2 protocol.
  - J1708 APIs - Read / write data with J1708 protocol.
  - J1587 APIs - Read / write data with J1587 protocol.
- **CONTROL PANEL**
  - CONTROL PANEL APIs - Core functions for controlling external smart display device.
  - BRIGHTNESS APIs - Configure brightness of external smart display.
  - HOTKEY APIs - Control / configure hot keys of external smart display.
  - LIGHT SENSOR APIs - Read data from light sensor.
- **DIO (Digital Input/Output)**

  - DIO APIs - Read / write status of DIO pins.
- **IIC**
  - IIC APIs - Read / write I2C data. Get bus status.
- **EEPROM**
  - EEPROM APIs - Read / write data to EEPROM. Get EEPROM status.
- **DISPLAY**
  - DISPLAY APIs - Control embedded display.

3.1.2 Android

The following figure describes the software stack of TREK SDK:



The SDK is composed of the following function domains:

- **SUSI**

  A VPM(Vehicle Power Management) MCU(Micro Controller Unit) is embedded in the device, which controls the power status of device and peripheral devices such as G-Sensor and P-Sensors.

  The SUSI function domain is designed in client-service architecture. The SUSI Service acts as a proxy to serve multiple APP processes simultaneously for accessing the VPM MCU and controlling the core functions and power management related functions of the TREK device.

  In your APP, you can use the following APIs exported in the TREK Client APIs Library to communicate with the SUSI service.

  SUSI Service Client API Modules:

  - **SUSI SRV**
    - SUSI SRV - APIs to connect client application process with SUSI Service or query status of SUSI Service.
  - **CORE**
    - CORE APIs - SDK core functions. Reset device, get status and configure system registry.
  - **VPM (Vehicle Power Management)**
    - VPM APIs - VPM related functions such as boot control, Ignition control, event delay adjustments, low battery protection and ETC..
    - WATCHDOG APIs - Simple APIs for you to construct your hardware revive mechanism.
    - PERIPHERAL CONTROL APIs - Control/configure peripheral devices.
    - G-SENSOR APIs - Read data from Gravity sensor.
  - **DIO (Digital Input/Output)**
    - DIO APIs - Read / write status of DIO pins.
  - **EEPROM**
    - EEPROM APIs - Read / write data to EEPROM. Get EEPROM status.

- **SDP** (Smart Display Panel)

  Depends on the specific device spec, the device may bundle with a smart display panel module. The

smart display panel module is embedded with a MCU to control functions of the module. Similar with IVCP function domain, the SDP function domain is also designed in client-service architecture. You can use the SDP APIs exported in the TREK Client APIs Library to communicate with the SDP service.

SDP Service Client API Modules:

- **SDP SRV**
  - SDP SRV - APIs to connect client application process with SDP Service or query status of SDP Service.
- **CONTROL PANEL**
  - CONTROL PANEL APIs - Core functions for controlling external smart display device.
  - BRIGHTNESS APIs - Configure brightness of external smart display.
  - HOTKEY APIs - Control / configure hot keys of external smart display.
  - LIGHT SENSOR APIs - Read data from light sensor.

- **VCIL** (Vehicle Communication Interface Layer)
  A VCIM(**V**ehicle **C**ommunication **I**nterface **M**odule) MCU is embedded in the device for controlling the vehicle communication protocols (e.g. CAN, J1939, OBD2, J1708, J1587). For the performance considerations, the VCIL function domain is designed in form of libraries, You can use the VCIL APIs exported in the VCIL API Library to control the MCU directly. For VCIL does not has service layer, the VCIL API Library does **NOT** support multi-process access.

  VCIL API Modules:

  - **VCIL**
    - VCIL APIs - Configuration physical channel for specific protocol.
    - CAN APIs - Read / write data with CAN protocol.
    - J1939 APIs - Read / write data with J1939 protocol.
    - OBD2 APIs - Read / write data with OBD2 protocol.
    - J1708 APIs - Read / write data with J1708 protocol.
    - J1587 APIs - Read / write data with J1587 protocol.

3.2 System Requirement

3.2.1 Supported Hardware

- TREK-520
- TREK-674
- TREK-572
- TREK-570

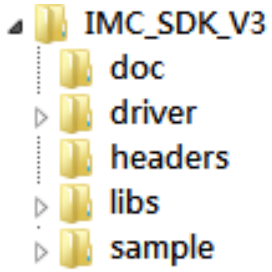
3.2.2 Supported Platforms

- WinCE 6.0 (TREK-520 only)
- Linux x86
- Windows 7/8/10
- Android (TREK-570 only)

3.3 SDK Package Structure

3.3.1 Window/Linux

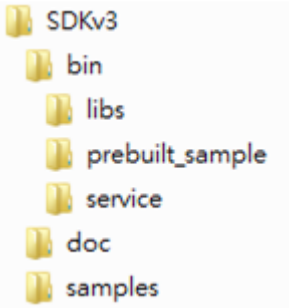
You can get the following contents by extracting the SDK package:



Directory	Description
doc	Document and Release Note
driver	SDK driver
headers	The library C header file
libs	The SDK library files for each supported platform.
sample	The SDK sample codes for each supported platform.
bin	The prebuilt binaries and dynamic libraries for each supported platform.

3.3.2 Android

You can get the following contents by extracting the SDK package:



Files/Directories	Description
bin/library/	The Java library and native library files. These libraries should be imported in to your APP project.
bin/service/	The TREK SDK service APK file. The service APK file must be installed into your device before running your APP or prebuilt sample APPs.
bin/prebuilt_sample/	The prebuilt APK files of sample codes.
doc	The Documents.
samples	The sample code.

4 SDK installation

4.1 Android

For that the TREK SDK for Android is designed in Client-Service structure, you need to install the TREK SDK Services to your TREK device and import the service client API libraries to your APP project.
The following sections describe how to install SDK.

4.1.1 Install TREK SDK Services

Please follow the steps to install TREK SDK Services to TREK device

1. Find the following files in the SDK package directory **/bin/service/**
    - **_install_mrm_service.bat**
    - **mrm_service.apk**
    - **susi_config**
    - **vcil_config**

2. Connect your TREK device to your PC though ADB

3. Execute **_install_mrm_service.bat** to do the install commands.

4.1.2 Import TREK SDK Client API

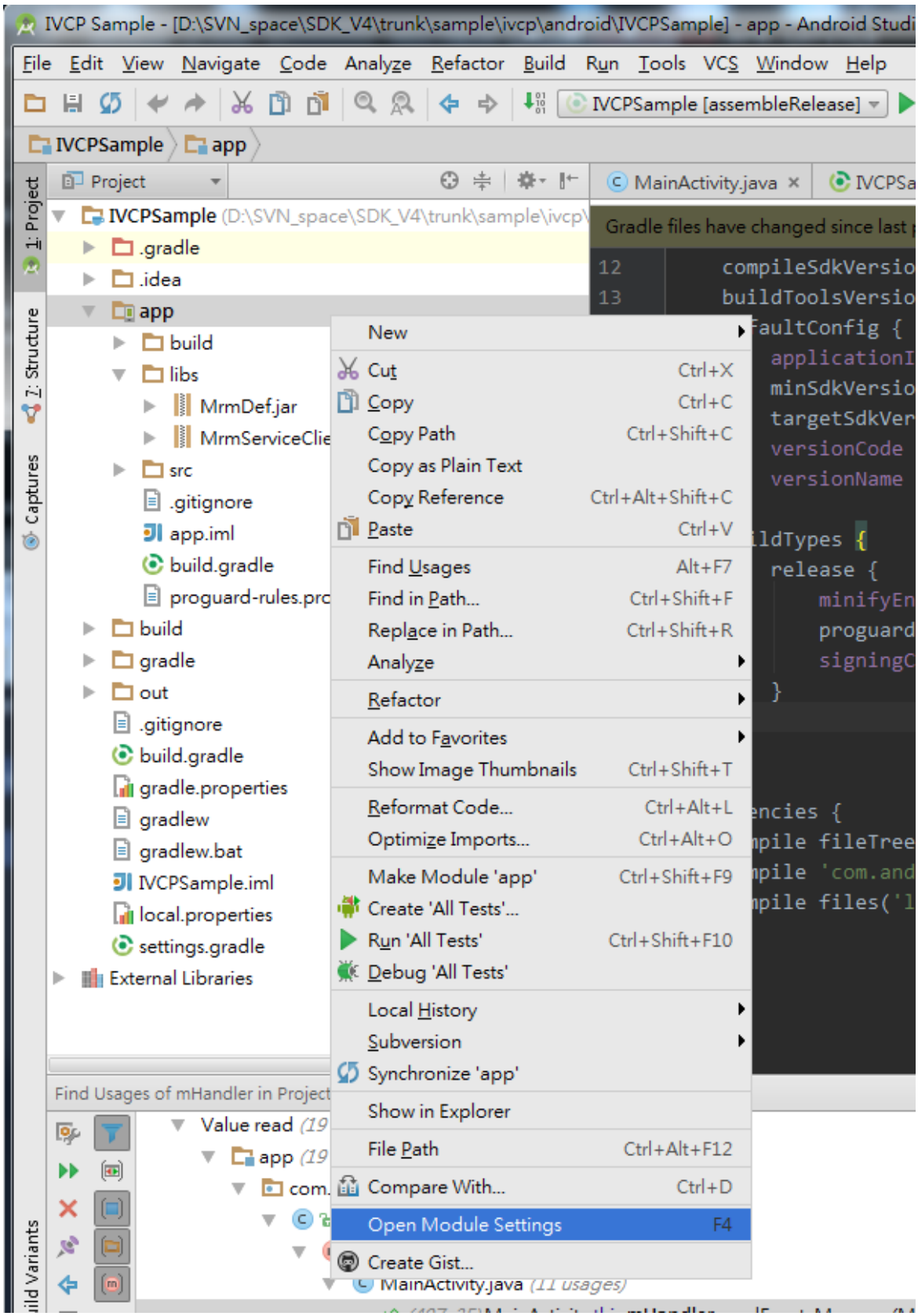
To access TREK SDK Services, you must import the Client API libs to your APP project.
The following steps shows how to import TREK SDK Client API libraries to an Android Studio APP project.

1. **Import TREK SDK Client API Libraries**
   (NOTE: This step is only necessary for SUSI and SDP function. You can skip this step if you only need VCIL functions)

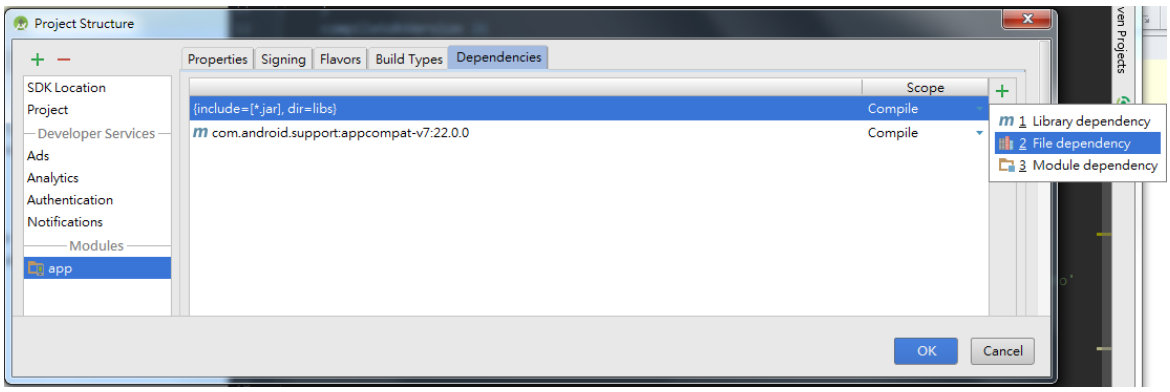
   - Find the following files in the SDK package directory **/libs**
     - **MrmServiceClientAPI.jar**
     - **MrmDef.jar**

   - Copy the libraries to the directory **/[Module Name]/libs/** in you Android Studio project (the default module name might be "app").

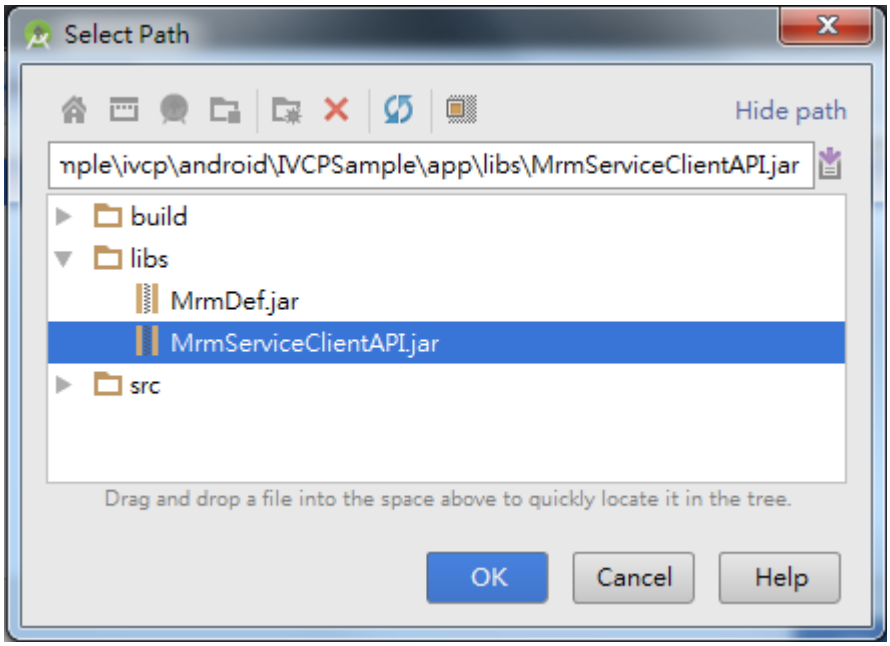
   - Right click on you APP module. Click "**Open module settings**"



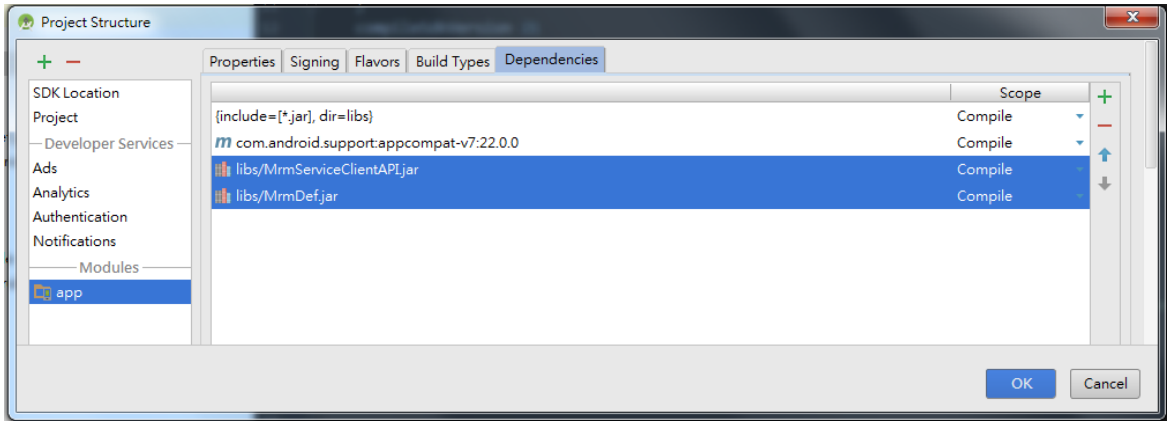
- Click the "**Dependency**" tab. Then click "**+**" -> "**File dependency**"



○ Select the library file.



○ Repeat the above steps to add all libs and you will see all libs are added to the list.

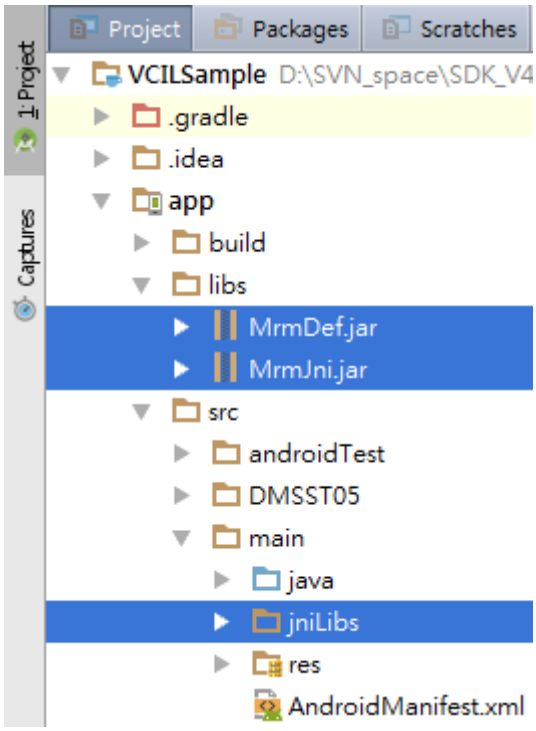


2. **Import VCIL APIs Libraries**

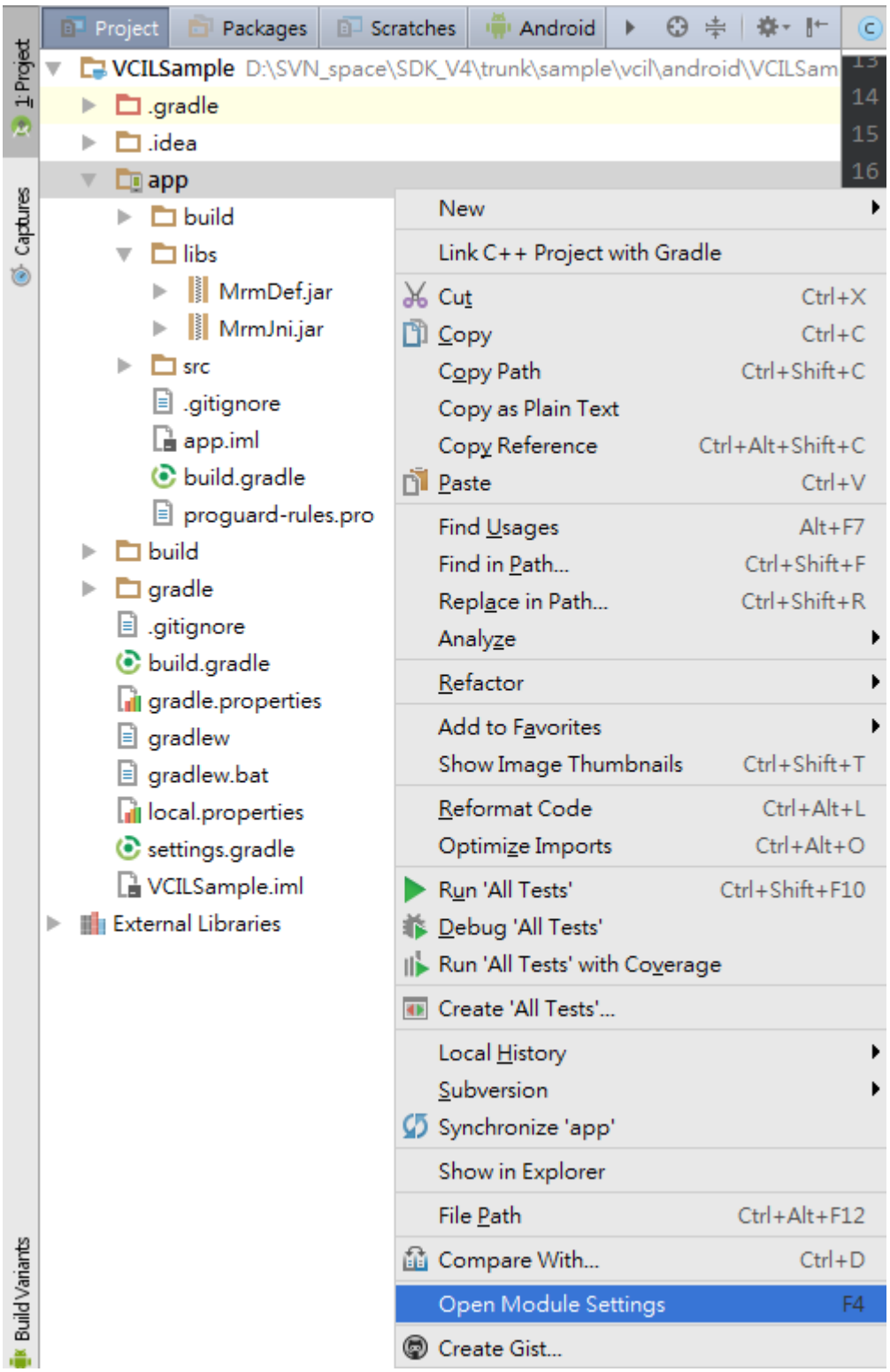
   (NOTE: This step is only necessary for VCIL functions)

   To access VCIL functions from your APP, you must import the VCIL libraries into you project.

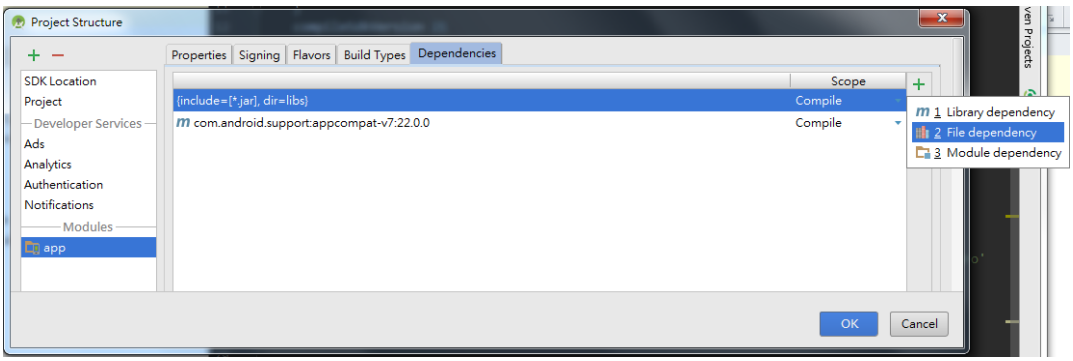
   - Please find the **MrmJni.jar, MrmDef.jar** and **jniLibs/** folder in the MRM SDK package. Copy the **MrmJni.jar, MrmDef.jar** to the directory **/[Module Name]/libs/** in your Android Studio project (the default module name might be "app") and copy the **jniLibs/** folder to the directory **/[Module Name]/src/main/** .



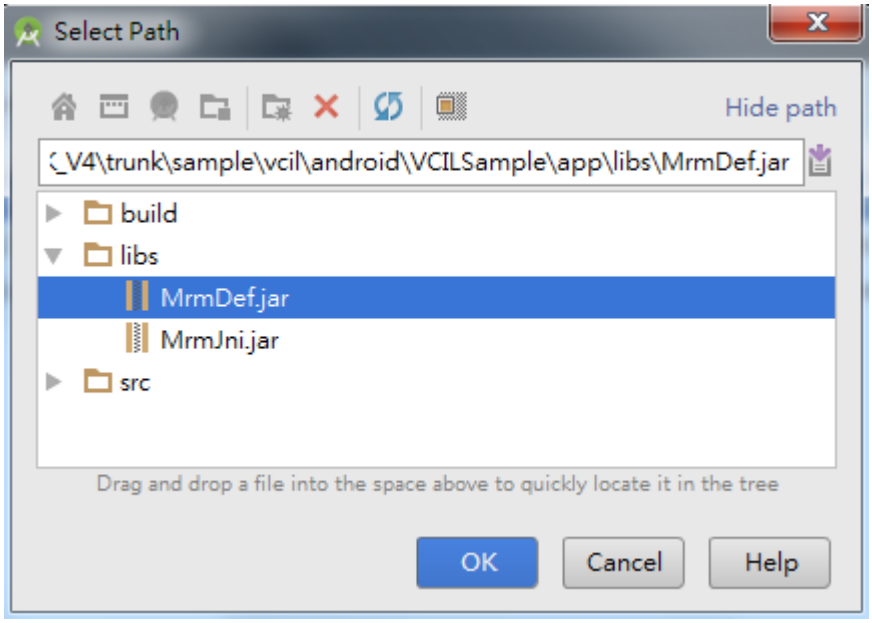
   - Then import the Java libraries by following the steps below:
     - Right click on you APP module. Click "**Open module settings**"



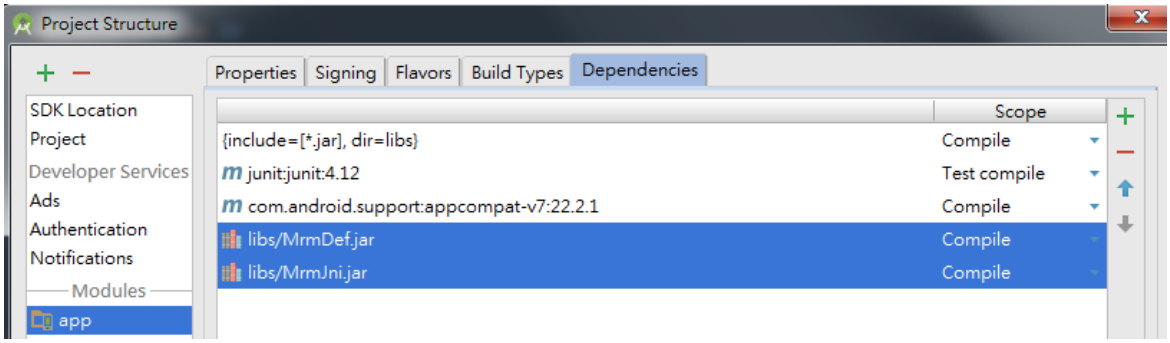
- Click the "**Dependency**" tab. Then click "+" -> "**File dependency**"



- Select the lib file.



- Repeat the above steps to add all libs and you will see all libs are added to the list.



5 Application Programming Interface (API)

5.1 Library Mapping

The following table describes the relationship of APIs and the corresponding libraries, and also, describes the platform support status.

Library Name	API Name	Support CE	Support Linux	Support Win	Support Android
Windows: - Linux : - Android: MrmServiceClientAPI.jar mrm_service.apk	SUSI_IMC_SUSISRV_BindService	X	X	X	○
	SUSI_IMC_SUSISRV_UnbindService	X	X	X	○
	SUSI_IMC_SUSISRV_IsServiceInitialized	X	X	X	○
	SUSI_IMC_SUSISRV_IsServiceConnected	X	X	X	○
Windows: - Linux : - Android: MrmServiceClientAPI.jar mrm_service.apk	SUSI_IMC_VCILSRV_BindService	X	X	X	○
	SUSI_IMC_VCILSRV_UnbindService	X	X	X	○
	SUSI_IMC_VCILSRV_IsServiceInitialized	X	X	X	○
	SUSI_IMC_VCILSRV_IsServiceConnected	X	X	X	○
Windows: - Linux : - Android: MrmServiceClientAPI.jar mrm_service.apk	SUSI_IMC_SDPSRV_BindService	X	X	X	○
	SUSI_IMC_SDPSRV_UnbindService	X	X	X	○
	SUSI_IMC_SDPSRV_IsServiceInitialized	X	X	X	○
	SUSI_IMC_SDPSRV_IsServiceConnected	X	X	X	○
Windows: SUSI_IMC_CORE_FU NCTION.dll Linux : libsusi.so Android: MrmServiceClientAPI.jar mrm_service.apk	SUSI_IMC_CORE_Init	X	○	X	X
	SUSI_IMC_CORE_DeInit	X	○	X	X
	SUSI_IMC_CORE_Available	○	○	X	X
	SUSI_IMC_CORE_GetLibVersion	○	○	X	X
	SUSI_IMC_CORE_GetPlatformName	○	○	X	○
	SUSI_IMC_CORE_AccessBootCounter	○	X	X	X
	SUSI_IMC_CORE_SoftReset	○	X	X	X
	SUSI_IMC_CORE_HardReset	○	X	X	X

	SUSI_IMC_CORE_Reset	X	○	X	X
	SUSI_IMC_CORE_GetImageInfo	○	○	X	○
	SUSI_IMC_CORE_RegistrySave	○	X	X	X
	SUSI_IMC_CORE_RegistryClean	○	X	X	X
Windows : SUSI_IMC_DIO.dll Linux : libsusi.so Android: MrmServiceClientAPI.jar mrm_service.apk	SUSI_IMC_DIO_GetLibVersion	○	X	○	X
	SUSI_IMC_DIO_Initialize	○	○	○	X
	SUSI_IMC_DIO_Deinitialize	○	○	○	X
	SUSI_IMC_DIO_ReadDIPin	○	○	○	○
	SUSI_IMC_DIO_WriteDOPin	○	○	○	○
	SUSI_IMC_DIO_GetVoiceSourceControlPin	X	△	○	X
	SUSI_IMC_DIO_SetVoiceSourceControlPin	X	△	○	X
	SUSI_IMC_DIO_GetBluetoothEnablePin	X	△	○	X
	SUSI_IMC_DIO_SetBluetoothEnablePin	X	△	○	X
	SUSI_IMC_DIO_GetVoiceModuleSelectPin	X	△	○	X
	SUSI_IMC_DIO_SetVoiceModuleSelectPin	X	△	○	X
Windows : SUSI_IMC_IIC.dll Linux : libsusi.so Android: MrmServiceClientAPI.jar mrm_service.apk	SUSI_IMC_IIC_GetLibVersion	○	X	X	X
	SUSI_IMC_IIC_Available	○	※	X	X
	SUSI_IMC_IIC_Read	○	※	X	X
	SUSI_IMC_IIC_Write	○	※	X	X
	SUSI_IMC_IIC_WriteReadCombine	○	※	X	X
Windows: SUSI_IMC_SMBUS.dll Linux: libsusi.so Android: MrmServiceClientAPI.jar mrm_service.apk	SUSI_IMC_EEPROM_Initialize	X	△	○	X
	SUSI_IMC_EEPROM_Deinitialize	X	△	○	X
	SUSI_IMC_EEPROM_GetLibVersion	X	△	○	○
	SUSI_IMC_EEPROM_GetEEPROMSize	X	△	○	○
	SUSI_IMC_EEPROM_ReadByte	X	△	○	○
	SUSI_IMC_EEPROM_WriteByte	X	△	○	○
	SUSI_IMC_EEPROM_ReadMultiByte	X	△	○	○
	SUSI_IMC_EEPROM_WriteMultiByte	X	△	○	○
Windows : SUSI_IMC_DISPLAY.dll Linux : libsusi.so	SUSI_IMC_DISPLAY_GetLibVersion	○	X	X	X
	SUSI_IMC_DISPLAY_Available	○	○	X	X
	SUSI_IMC_DISPLAY_ScreenOn	○	○	X	X
	SUSI_IMC_DISPLAY_ScreenOff	○	○	X	X

Android: MrmServiceClientAPI.jar mrm_service.apk	SUSI_IMC_DISPLAY_GetBrightRange	○	○	X	X
	SUSI_IMC_DISPLAY_GetBright	○	○	X	X
	SUSI_IMC_DISPLAY_SetBright	○	○	X	X
Windows : SUSI_IMC_LIGHT_SENSOR.dll Linux : libsusi.so Android: MrmServiceClientAPI.jar mrm_service.apk	SUSI_IMC_LIGHTSENSOR_GetLibVersion	○	X	X	○
	SUSI_IMC_LIGHTSENSOR_GetStatus	○	○	X	○
Windows : SUSI_IMC_WATCH_DOG.dll Linux : libsusi.so Android: MrmServiceClientAPI.jar mrm_service.apk	SUSI_IMC_WD_Initialize	○	○	○	X
	SUSI_IMC_WD_Deinitialize	○	○	○	X
	SUSI_IMC_WD_Trigger	○	○	○	○
	SUSI_IMC_WD_Enabled	○	○	○	○
	SUSI_IMC_WD_GetTime	○	○	○	○
	SUSI_IMC_WD_SetTime	○	○	○	○
	SUSI_IMC_WD_GetRange	○	○	○	○
	SUSI_IMC_WD_GetLibVersion	○	X	○	X
Windows : SUSI_IMC_PeripheralCtrl.dll Linux : libsusi.so Android: MrmServiceClientAPI.jar mrm_service.apk	SUSI_IMC_PERIPHERALCTRL_Initialize	○	○	○	X
	SUSI_IMC_PERIPHERALCTRL_Deinitialize	○	○	○	X
	SUSI_IMC_PERIPHERALCTRL_GetLibVersion	○	X	○	X
	SUSI_IMC_PERIPHERALCTRL_SetPeripheralControl	○	○	○	○
	SUSI_IMC_PERIPHERALCTRL_GetPeripheralControl	○	○	○	○
	SUSI_IMC_PERIPHERALCTRL_SetRearViewSource	X	○	○	○
	SUSI_IMC_PERIPHERALCTRL_GetRearViewSource	X	○	○	○
	SUSI_IMC_PERIPHERALCTRL_SetAutoRearView	X	○	○	○
	SUSI_IMC_PERIPHERALCTRL_GetAutoRearView	X	○	○	○
Windows : SUSI_IMC_GSensor.dll Linux : libsusi.so Android: MrmServiceClientA	SUSI_IMC_GSENSOR_Initialize	○	○	○	X
	SUSI_IMC_GSENSOR_Deinitialize	○	○	○	X
	SUSI_IMC_GSENSOR_GetLibVersion	○	X	○	X
	SUSI_IMC_GSENSOR_GetData	○	○	○	○
	SUSI_IMC_GSENSOR_GetResolution	○	○	○	○

PI.jar mrm_service.apk	SUSI_IMC_GSENSOR_SetResolution	○	○	○	○
	SUSI_IMC_GSENSOR_AlarmEnabled	○	○	○	○
	SUSI_IMC_GSENSOR_SetAlarmEvent	○	○	○	X
	SUSI_IMC_GSENSOR_SetAlarmEvent Handler	X	X	X	○
	SUSI_IMC_GSENSOR_UnsetAlarmEve ntHandler	X	X	X	○
	SUSI_IMC_GSENSOR_WaitAlarmEvent	X	X	X	○
	SUSI_IMC_GSENSOR_GetAlarmData	○	○	○	○
	SUSI_IMC_GSENSOR_GetAlarmThres hold	○	○	○	○
	SUSI_IMC_GSENSOR_SetAlarmThresh old	○	○	○	○
	SUSI_IMC_GSENSOR_GetWakeupThr eshold	○	○	○	○
	SUSI_IMC_GSENSOR_SetWakeupThre shold	○	○	○	○
Windows : SUSI_IMC_VPM.dll Linux : libsusi.so Android: MrmServiceClientA PI.jar mrm_service.apk	SUSI_IMC_VPM_Initialize	○	○	○	X
	SUSI_IMC_VPM_Deinitialize	○	○	○	X
	SUSI_IMC_VPM_GetFWVersion	○	○	○	○
	SUSI_IMC_VPM_GetLibVersion	○	X	○	X
	SUSI_IMC_VPM_SetIgnitionMode	○	○	X	○
	SUSI_IMC_VPM_GetIgnitionMode	○	○	X	○
	SUSI_IMC_VPM_SetIgnOffEventDelay	○	○	○	○
	SUSI_IMC_VPM_GetIgnOffEventDelay	○	○	○	○
	SUSI_IMC_VPM_SetIgnHardOffDelay	○	○	○	○
	SUSI_IMC_VPM_GetIgnHardOffDelay	○	○	○	○
	SUSI_IMC_VPM_SetIgnSuspendHardOf fDelay	○	○	X	○
	SUSI_IMC_VPM_GetIgnSuspendHardO ffDelay	○	○	X	○
	SUSI_IMC_VPM_SetIgnOnDelay	○	○	○	○
	SUSI_IMC_VPM_GetIgnOnDelay	○	○	○	○
	SUSI_IMC_VPM_SetLBPLowDelay	○	○	○	○
	SUSI_IMC_VPM_GetLBPLowDelay	○	○	○	○
	SUSI_IMC_VPM_SetLBPLowHardDelay	○	○	○	○
	SUSI_IMC_VPM_GetLBPLowHardDelay	○	○	○	○
	SUSI_IMC_VPM_SetLBPThresholdFor1 2VSystem	○	○	○	○

	SUSI_IMC_VPM_GetLBPThresholdFor12VSystem	o	o	o	o
	SUSI_IMC_VPM_SetLBPThresholdFor24VSystem	o	o	o	o
	SUSI_IMC_VPM_GetLBPThresholdFor24VSystem	o	o	o	o
	SUSI_IMC_VPM_SetPrebootThresholdFor12VSystem	o	o	o	o
	SUSI_IMC_VPM_GetPrebootThresholdFor12VSystem	o	o	o	o
	SUSI_IMC_VPM_SetPrebootThresholdFor24VSystem	o	o	o	o
	SUSI_IMC_VPM_GetPrebootThresholdFor24VSystem	o	o	o	o
	SUSI_IMC_VPM_GetLBPThresholdInfo	o	o	o	o
	SUSI_IMC_VPM_GetIgnitionStatus	o	o	o	o
	SUSI_IMC_VPM_GetCarBatteryVoltage	o	o	o	o
	SUSI_IMC_VPM_GetCurrentCarBatteryMode	o	o	o	o
	SUSI_IMC_VPM_SetLBPStatus	o	o	o	o
	SUSI_IMC_VPM_GetLBPStatus	o	o	o	o
	SUSI_IMC_VPM_SetPrebootLBPStatus	o	o	o	o
	SUSI_IMC_VPM_GetPrebootLBPStatus	o	o	o	o
	SUSI_IMC_VPM_SetATMode	o	o	o	o
	SUSI_IMC_VPM_GetATMode	o	o	o	o
	SUSI_IMC_VPM_SetKeepAliveMode	o	o	o	o
	SUSI_IMC_VPM_GetKeepAliveMode	o	o	o	o
	SUSI_IMC_VPM_SetWakeupSourceControlStatus	o	o	o	o
	SUSI_IMC_VPM_GetWakeupSourceControlStatus	o	o	o	o
	SUSI_IMC_VPM_GetLastWakeupSource	o	o	o	o
	SUSI_IMC_VPM_LoadDefaultValue	o	o	o	o
	SUSI_IMC_VPM_SetAlarmActive	o	o	o	o
	SUSI_IMC_VPM_GetAlarmActive	o	o	o	o
	SUSI_IMC_VPM_SetAlarmTime	o	o	o	o
	SUSI_IMC_VPM_GetAlarmTime	o	o	o	o
	SUSI_IMC_VPM_GetBackupBatteryVoltage	o	o	**X**	o
	SUSI_IMC_VPM_GetBackupBatteryRemainingCapacity	o	o	**X**	o
	SUSI_IMC_VPM_GetBackupBatteryPercentageOfCharge	o	o	**X**	o

	SUSI_IMC_VPM_GetBackupBatteryTemperature	○	○	X	○
	SUSI_IMC_VPM_GetBackupBatteryRemainingTime	○	○	X	○
	SUSI_IMC_VPM_GetBackupBatteryTimeToFull	○	○	X	○
	SUSI_IMC_VPM_GetBackupBatteryInfo	○	○	X	○
	SUSI_IMC_VPM_SetRTCTime	X	○	○	○
	SUSI_IMC_VPM_GetRTCTime	X	○	○	○
Windows : SUSI_IMC_VCIL.dll Linux : libvcil.so Android: MrmServiceClientAPI.jar mrm_service.apk	SUSI_IMC_VCIL_Initialize	○	○	○	X
	SUSI_IMC_VCIL_InitializeEx	○	○	○	X
	SUSI_IMC_VCIL_Deinitialize	○	○	○	X
	SUSI_IMC_VCIL_GetLibVersion	○	○	○	X
	SUSI_IMC_VCIL_GetFWVersion	○	○	○	○
	SUSI_IMC_VCIL_ResetModule	○	○	○	○
	SUSI_IMC_VCIL_ModuleControl	○	○	○	○
Windows : SUSI_IMC_VCIL.dll Linux : libvcil.so Android: MrmServiceClientAPI.jar mrm_service.apk	SUSI_IMC_CAN_Read	○	○	○	○
	SUSI_IMC_CAN_ReadMulti	X	X	X	○
	SUSI_IMC_CAN_Write	○	○	○	○
	SUSI_IMC_CAN_SetEvent	○	○	○	X
	SUSI_IMC_CAN_SetEventHandler	X	X	X	○
	SUSI_IMC_CAN_UnsetEventHandler	X	X	X	○
	SUSI_IMC_CAN_WaitEvent	X	X	X	○
	SUSI_IMC_CAN_SetBitTiming	○	○	○	○
	SUSI_IMC_CAN_SetMessageMask	○	○	○	○
	SUSI_IMC_CAN_GetMessageMask	○	○	○	○
	SUSI_IMC_CAN_RemoveMessageMask	○	○	○	○
	SUSI_IMC_CAN_ResetMessageMask	○	○	○	○
Windows : SUSI_IMC_VCIL.dll Linux : libvcil.so Android: MrmServiceClientAPI.jar mrm_service.apk	SUSI_IMC_J1708_Read	○	○	○	○
	SUSI_IMC_J1708_ReadMulti	X	X	X	○
	SUSI_IMC_J1708_Write	○	○	○	○
	SUSI_IMC_J1708_SetEvent	○	○	○	X
	SUSI_IMC_J1708_SetEventHandler	X	X	X	○
	SUSI_IMC_J1708_UnsetEventHandler	X	X	X	○

	SUSI_IMC_J1708_WaitEvent	X	X	X	○
	SUSI_IMC_J1708_AddMessageFilter	○	○	○	○
	SUSI_IMC_J1708_GetMessageFilter	○	○	○	○
	SUSI_IMC_J1708_RemoveMessageFilter	○	○	○	○
	SUSI_IMC_J1708_RemoveAllMessageFilter	○	○	○	○
Windows : SUSI_IMC_VCIL.dll Linux : libvcil.so Android: MrmServiceClientAPI.jar mrm_service.apk	SUSI_IMC_J1587_Read	○	○	○	○
	SUSI_IMC_J1587_ReadMulti	X	X	X	○
	SUSI_IMC_J1587_Write	○	○	○	○
	SUSI_IMC_J1587_SetEvent	○	○	○	X
	SUSI_IMC_J1587_SetEventHandler	X	X	X	○
	SUSI_IMC_J1587_UnsetEventHandler	X	X	X	○
	SUSI_IMC_J1587_WaitEvent	X	X	X	○
	SUSI_IMC_J1587_AddMessageFilter	○	○	○	○
	SUSI_IMC_J1587_GetMessageFilter	○	○	○	○
	SUSI_IMC_J1587_RemoveMessageFilter	○	○	○	○
	SUSI_IMC_J1587_RemoveAllMessageFilter	○	○	○	○
Windows : SUSI_IMC_VCIL.dll Linux : libvcil.so Android: MrmServiceClientAPI.jar mrm_service.apk	SUSI_IMC_J1939_Read	○	○	○	○
	SUSI_IMC_J1939_ReadMulti	X	X	X	○
	SUSI_IMC_J1939_Write	○	○	○	○
	SUSI_IMC_J1939_SetEvent	○	○	○	X
	SUSI_IMC_J1939_SetEventHandler	X	X	X	○
	SUSI_IMC_J1939_UnsetEventHandler	X	X	X	○
	SUSI_IMC_J1939_WaitEvent	X	X	X	○
	SUSI_IMC_J1939_AddMessageFilter	○	○	○	○
	SUSI_IMC_J1939_GetMessageFilter	○	○	○	○
	SUSI_IMC_J1939_RemoveMessageFilter	○	○	○	○
	SUSI_IMC_J1939_SetTransmitConfig	○	○	○	○
	SUSI_IMC_J1939_GetTransmitConfig	○	○	○	○
Windows : SUSI_IMC_VCIL.dll Linux : libvcil.so	SUSI_IMC_OBD2_Read	○	○	○	○
	SUSI_IMC_OBD2_ReadMulti	X	X	X	○
	SUSI_IMC_OBD2_Write	○	○	○	○

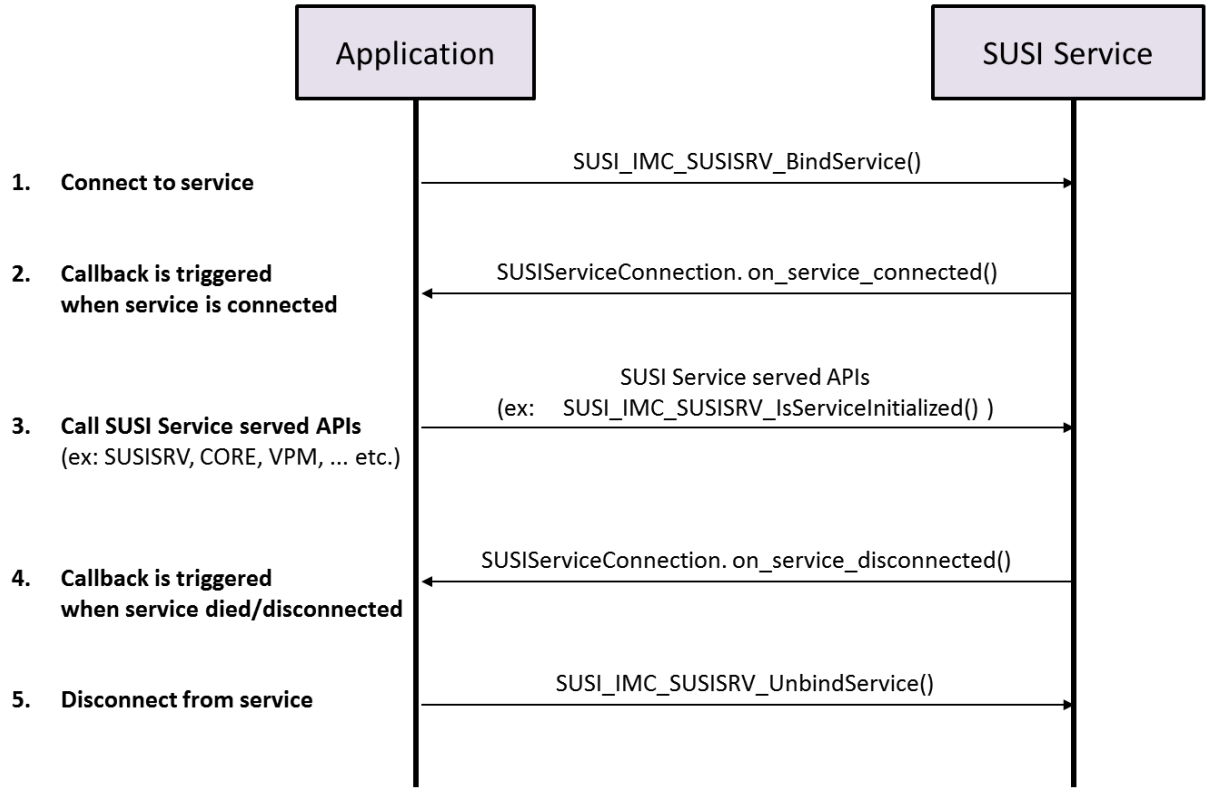
Android: MrmServiceClientAPI.jar mrm_service.apk	SUSI_IMC_OBD2_SetEvent	○	○	○	X
	SUSI_IMC_OBD2_SetEventHandler	X	X	X	○
	SUSI_IMC_OBD2_UnsetEventHandler	X	X	X	○
	SUSI_IMC_OBD2_WaitEvent	X	X	X	○
	SUSI_IMC_OBD2_AddMessageFilter	○	○	○	○
	SUSI_IMC_OBD2_GetMessageFilter	○	○	○	○
	SUSI_IMC_OBD2_RemoveMessageFilter	○	○	○	○
	SUSI_IMC_OBD2_RemoveAllMessageFilter	○	○	○	○
Windows : SUSI_IMC_CONTROLPANEL.dll CSerialPort.dll Linux: libSUSI_IMC_CONTROLPANEL.so Android: MrmServiceClientAPI.jar mrm_service.apk	SUSI_IMC_CONTROLPANEL_GetVersion	X	○	○	X
	SUSI_IMC_CONTROLPANEL_Initialize	X	○	○	X
	SUSI_IMC_CONTROLPANEL_Deinitialize	X	○	○	X
	SUSI_IMC_CONTROLPANEL_GetFirmwareInformation	X	○	○	○
	SUSI_IMC_CONTROLPANEL_ResetFirmware	X	○	○	○
	SUSI_IMC_BRIGHTNESS_SetDutyCycle	X	○	○	○
	SUSI_IMC_BRIGHTNESS_GetDutyCycle	X	○	○	○
	SUSI_IMC_BRIGHTNESS_SetLevel	X	○	○	○
	SUSI_IMC_BRIGHTNESS_GetLevel	X	○	○	○
	SUSI_IMC_LIGHTSENSOR_GetStatus	X	○	○	○
	SUSI_IMC_HOTKEY_Read	X	○	○	○
	SUSI_IMC_HOTKEY_RegisterOnPressEventMonitor	X	○	○	○
	SUSI_IMC_HOTKEY_ClearOnPressEventMonitor	X	○	○	○
	SUSI_IMC_HOTKEY_EnableEventMonitor	X	○	○	○
	SUSI_IMC_HOTKEY_DisableEventMonitor	X	○	○	○
	SUSI_IMC_HOTKEY_SetLedDutyCycle	X	○	○	○
	SUSI_IMC_HOTKEY_GetLedDutyCycle	X	○	○	○
Symbol Meaning	○ Fully supported				
	X Not supported				
	※ Already supported it for Linux ARM, but not supported on Linux x86.				
	□ Already supported it for Linux ARM, but on Linux x86 not ready yet.				
	△ Not supported it for Linux ARM, but supported on x86.				

5.2 SUSI SRV

5.2.1 Usage Overview

In TREK SDK for Android, the CORE, VPM, WATCHDOG, G-SENSOR and DIO API module are served by SUSI Service. Before using these API, you must connect your APP process to the SUSI Service.
The SUSISRV API module provides APIs to connect/disconnect APP to/from SUSI service, and also APIs for getting status of SUSI service.
The following section describes the usage of APIs.

5.2.1.1 Android



The class for SUSISRV APIs for Android is **mrm.client.SUSIServiceClient**.

To use APIs served by SUSI Service (ex: SUSISRV, CORE, VPM, ... etc.) , you must first call SUSI_IMC_SUSISRV_BindService() to connect your client application process to SUSI Service.
As the argument of SUSI_IMC_SUSISRV_BindService(), you must create a instance of class - SUSIServiceConnection and implement the methods:

- **on_service_connected**() and
- **on_service_disconnected**()

to define the behavior of your application's behavior when service is connected and disconnected.

After **SUSI_IMC_SUSISRV_BindService**() is called, the client application will try to connect to the SUSI Service running in background. If SUSI Service is not running, the service will be started and all API modules served by SUSI Service will be initialized.
If client application process is connected to service correctly, the implemented **SUSIServiceConnection**.**on_service_connected**() will be callback. After the callback, you can then start calling APIs served by SUSI Service (ex: SUSISRV, CORE, VPM, ... etc.).
If SUSI Service died or disconnected abnormally, the implemented **SUSIServiceConnection**.**on_service_disconnected**() will be callback so your application process will be informed.

To disconnect your application from SUSI service, you can call SUSI_IMC_SUSISRV_UnbindService() to release allocated resources in SUSI Service. You should always disconnect your application from service before the applications is closed.

For details, please refer to the sample code - SUSISample.

5.2.2 Structure/Classes

5.2.2.1 Android

SUSIServiceConnection

class:

mrm.client.**SUSIServiceConnection**

Fields:

-

Methods:

Syntax	Comment
abstract void **on_service_connected**()	The callback which will be triggered when client application process is connected to SUSI Service is successfully after SUSI_IMC_SUSISRV_BindService() is called. You must implement this method to define the your application's behavior when service connected.
abstract void **on_service_disconnected**()	The callback which will be triggered when SUSI Service is successfully died or disconnected abnormally. You must implement this method to define the your application's behavior when service disconnected.

Remark:

For details of implementation, please refer to the sample code - SUSISample.

5.2.3 APIs

5.2.3.1 SUSI_IMC_SUSISRV_BindService

Syntax:

Android	short **SUSI_IMC_SUSISRV_BindService**(SUSIServiceConnection conn)

Description:

Connect client application process to running SUSI Service,

If SUSI Service is not running when this API is called, the service will be started and initialized.

Parameters:

conn [in]

Instance of **SUSIServiceConnection**.

You must implement

SUSIServiceConnection.**on_service_connected**()/**on_service_disconnected**() to define the your application's behavior when service connected/disconnected.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Android

Remark:

-

5.2.3.2 SUSI_IMC_SUSISRV_UnbindService

Syntax:

Android	short **SUSI_IMC_SUSISRV_UnbindService**()

Description:

Disconnect client application process from SUSI Service and release allocated resources.

Parameters:

-

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Android

Remark:

-

5.2.3.3 SUSI_IMC_SUSISRV_IsServiceInitialized

Syntax:

Android	short **SUSI_IMC_SUSISRV_IsServiceInitialized**(boolean[] status)

Description:

Get initialization status of SUSI Service.

Parameters:

status

An allocated array of size 1 for storing returned initialization status.

The returned value will be store at index 0.

The value is TRUE, if SUSI Service is initialized and all served APIs are available.

The value is FALSE, if SUSI Service is not initialized.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Android

Remark:

-

5.2.3.4 SUSI_IMC_SUSISRV_IsServiceConnected

Syntax:

Android	boolean **SUSI_IMC_SUSISRV_IsServiceConnected**()

Description:

Check whether the client application process is connected to SUSI Service.

Parameters:

-

Returns:

Returns TRUE, if connected.

Returns FALSE, if disconnected.

Supported Platforms:

TREK-570 Android

Remark:

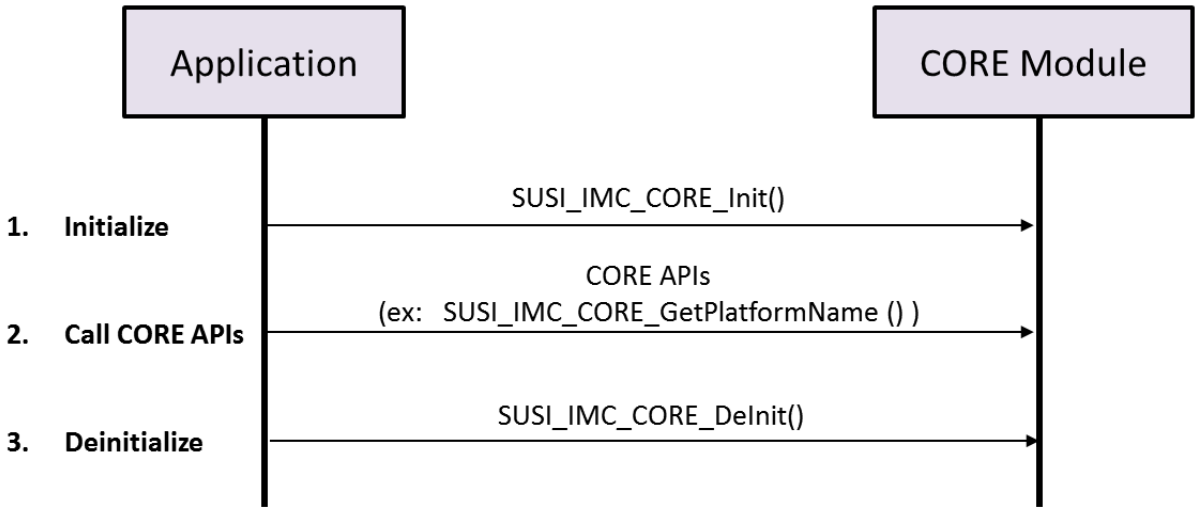
-

5.3 CORE

5.3.1 Usage Overview

The CORE API module provides APIs to get system informations and perform some system operations. The following section describes the usage of APIs.

5.3.1.1 Windows/Linux



To use CORE APIs. You need call SUSI_IMC_CORE_Init() to Initialize the CORE module first. Then you can start using CORE APIs.

To stop using the CORE module must call SUSI_IMC_CORE_DeInit() to close CORE module .

5.3.1.2 Android

The class for CORE APIs for Android is **mrm.client.SUSIServiceClient**.

For the basic usage, please refer to the section SUSI SRV Usage Overview.

5.3.2 Structure/Classes

5.3.2.1 Windows/Linux

IMC_CORE_GETPLATFORMNAME Structure

Syntax:

```
typedef struct
{
        TCHAR *PlatformName;
            DWORD *size;
}IMC_CORE_GETPLATFORMNAME,*PIMC_CORE_GETPLATFORMNAME;
```

Description:

This structure defines the Platform Name .

Members:

PlatformName [out]

Pointer to an array in which the platform name string is returned.

size [in/out]

Pointer to a variable that specifies the size, in TCHAR, of the array pointed to by the PlatformName parameter.

If PlatformName is given as NULL, when the function returns, the variable will contain the array size required for the platform name.

IMC_CORE_ACCESS_BOOTCOUNTER Structure

Syntax:

```
typedef struct
{
        DWORD mode;
            DWORD OPFlag;
            BOOL *enable;
            DWORD *value;
}IMC_CORE_ACCESS_BOOTCOUNTER,*PIMC_CORE_ACCESS_BOOTCOUNTER;
```

Description:

This structure defines the boot counter mode and it's status and value.

Members:

mode [in]

The value can be either

0 : get mode - To get information from counter.

1 : set mode - To set information to counter.

OPFlag [in]

The operation flag can be the combination of

1 : boot counter status - The operation is on the parameter enable.

2 : boot counter value - The operation is on the parameter value.

IMC_IMGINFO Structure

(CE version)
Syntax:

```
typedef struct
{
        TCHAR *WinCEImageVersion;
            TCHAR *WinCEBuiltDate;
            TCHAR *BootloaderVersion;
            TCHAR *BootloaderDate;
}IMC_IMGINFO, *PIMC_IMGINFO;
```

Description:

This structure defines the Windows CE Image/Bootloader version number and build date.

Members:

WinCEImageVersion

Windows CE version number

WinCEBuiltDate

Windows CE build date

BootloaderVersion

Bootloader version number

BootloaderDate

Bootloader build date

(Linux/Android version)
Syntax:

```
typedef struct
{
        TCHAR *AndroidVersion;
            TCHAR *AndroidBuiltDate;
            TCHAR *KernelVersion;
            TCHAR *KernelBuiltDate;
}IMC_IMGINFO, *PIMC_IMGINFO;
```

Description:

This structure defines the Android/Linux Kernel version number and build date.

Members:

AndroidVersion

Android version number

AndroidBuiltDate

Android build date

KernelVersion

Linux Kernel version number

KernelBuiltDate

Linux Kernel build date

5.3.2.2 Android

IMC_CORE_GETPLATFORMNAME

class:

mrm.define.core.**IMC_CORE_PLATFORMNAME**

Fields:

Field Name	Type	Input/Output	Comment
name	String	out	The platform name
size	long	out	The length of platform name

IMC_IMGINFO Structure

class:

mrm.define.core.**IMC_IMGINFO**

Fields:

Field Name	Type	Input/output	Comment
AndroidVersion	String	out	Android image version.
AndroidBuiltDate	String	out	Android built date information.
KernelVersion	String	out	Kernel version.
KernelBuiltDate	String	out	Kernel built date information.

5.3.3 APIs

5.3.3.1 SUSI_IMC_CORE_Init

Syntax:

Linux	USHORT **SUSI_IMC_CORE_Init**();

Description:

Initialize the IMC Library.

Parameters:

None

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux

TREK-570 Linux

TREK-572 Linux

TREK-674 Linux

Remark:

This API only supports Linux. CE and Windows both didn't need this API.

5.3.3.2 SUSI_IMC_CORE_DeInit

Syntax:

Linux	USHORT **SUSI_IMC_CORE_DeInit**();

Description:

Deinitialize the IMC Library.

Parameters:

None

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux

TREK-570 Linux

TREK-572 Linux

TREK-674 Linux

Remark:

This API only supports Linux. CE and Windows both didn't need this API.

5.3.3.3 SUSI_IMC_CORE_Available

Syntax:

Linux / CE	USHORT **SUSI_IMC_CORE_Available**();

Description:

Check Core driver is available.

Parameters:

None

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux

TREK-572 Linux

TREK-674 Linux

Remark:

None

5.3.3.4 SUSI_IMC_CORE_GetLibVersion

Syntax:

Linux / CE	USHORT **SUSI_IMC_CORE_GetLibVersion**(OUT BYTE version[18]);

Description:

Get library version.

Parameters:

version [out]

Point to read buffer that specifies the Core library's current version.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux

TREK-572 Linux

TREK-674 Linux

Remark:

The version buffer size must be greater than 18 bytes.

5.3.3.5 SUSI_IMC_CORE_GetPlatformName

Syntax:

Linux / CE	USHORT **SUSI_IMC_CORE_GetPlatformName**(PIMC_CORE_GETPLATFORMNAME parm);
Android	short **SUSI_IMC_CORE_GetPlatformName**(IMC_CORE_PLATFORMNAME parm)

Description:

Get the current platform name.

Parameters:

parm [in/out]

Linux / Windows:

Points to a variable that specifies the name of platform.

Android:

Instance of IMC_CORE_PLATFORMNAME for storing returned value;

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Android

TREK-572 Linux

TREK-674 Linux

Remark:

None

5.3.3.6 SUSI_IMC_CORE_AccessBootCounter

Syntax:

CE	USHORT **SUSI_IMC_CORE_AccessBootCounter**(PIMC_CORE_ACCESS_BOOTCOUNTER parm);

Description:

Access the boot counter. A boot counter is used to count the number of boot times.

Parameters:

parm [in/out]

Points to a variable that specifies the number of access boot counter.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 CE

Remark:

None

5.3.3.7 SUSI_IMC_CORE_SoftReset

Syntax:

CE	USHORT **SUSI_IMC_CORE_SoftReset**(void);

Description:

Software reset function.

Parameters:

None

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 CE

Remark:

Note: This API only supports CE. Linux didn't need this API.

5.3.3.8 SUSI_IMC_CORE_HardReset

Syntax:

CE	USHORT **SUSI_IMC_CORE_HardReset**(void);

Description:

Hardware reset function.

Parameters:

None

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 CE

Remark:

Note: This API only supports CE. Linux didn't need this API.

5.3.3.9 SUSI_IMC_CORE_Reset

Syntax:

Linux	USHORT **SUSI_IMC_CORE_Init**(void);

Description:

Hardware reset function.

Parameters:

None

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux

Remark:

Note: This API only supports Linux. CE didn't need this API.

5.3.3.10 SUSI_IMC_CORE_GetImageInfo

Syntax:

Linux / CE	USHORT **SUSI_IMC_CORE_GetImageinfo**(PIMC_IMGINFO pInfo);
Android	short **SUSI_IMC_CORE_GetImageinfo**(IMC_IMGINFO imgInfo)

Description:

Get image information, including WinCE image version, WinCE image build date, Boot loader image version, and Boot loader image build date.

Parameters:

pInfo [out]

Linux / Windows

Retrieve image information.

Android:

Instance of IMC_IMGINFO for storing returned value;

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-674 Linux

TREK-570 Linux / Android

TREK-572 Linux

Remark:

None

5.3.3.11 SUSI_IMC_CORE_RegistrySave

Syntax:

CE	USHORT **SUSI_IMC_CORE_RegistrySave**(void);

Description:

Save registry.

Parameters:

None

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 CE

Remark:

Note: This API only supports CE. Linux didn't need this API.

5.3.3.12 SUSI_IMC_CORE_RegistryClean

Syntax:

CE	USHORT **SUSI_IMC_CORE_RegistryClean**(void);

Description:

Clean registry to default.

Parameters:

None

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 CE

Remark:

Note: This API only supports CE. Linux didn't need this API.

5.4 VPM

5.4.1 Usage Overview

TREK device is equipped with Vehicle Power Manager (VPM) module which provide various power management functions such as ignition status monitoring, wakeup source settings, power mode settings, power management event delay settings, low battery protection and ETC.

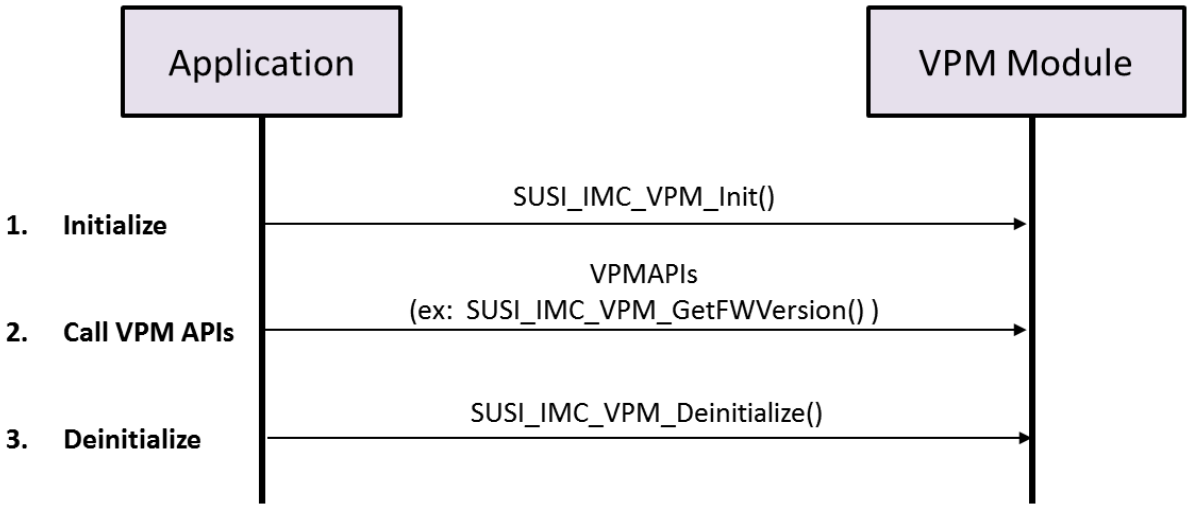
The VPM API module provides corresponding APIs for client application to access these functions.

The following parts describe the API usage of main VPM functions.

5.4.1.1 Basic Usage

This section describe the basic usage of VPM APIs for each platform.

Windows/Linux



For all VPM APIs, you should call SUSI_IMC_VPM_Initialize() before using the other VPM API. To stop the VPM module, you must use SUSI_IMC_VPM_Deinitialize() to deinitialize.

Android

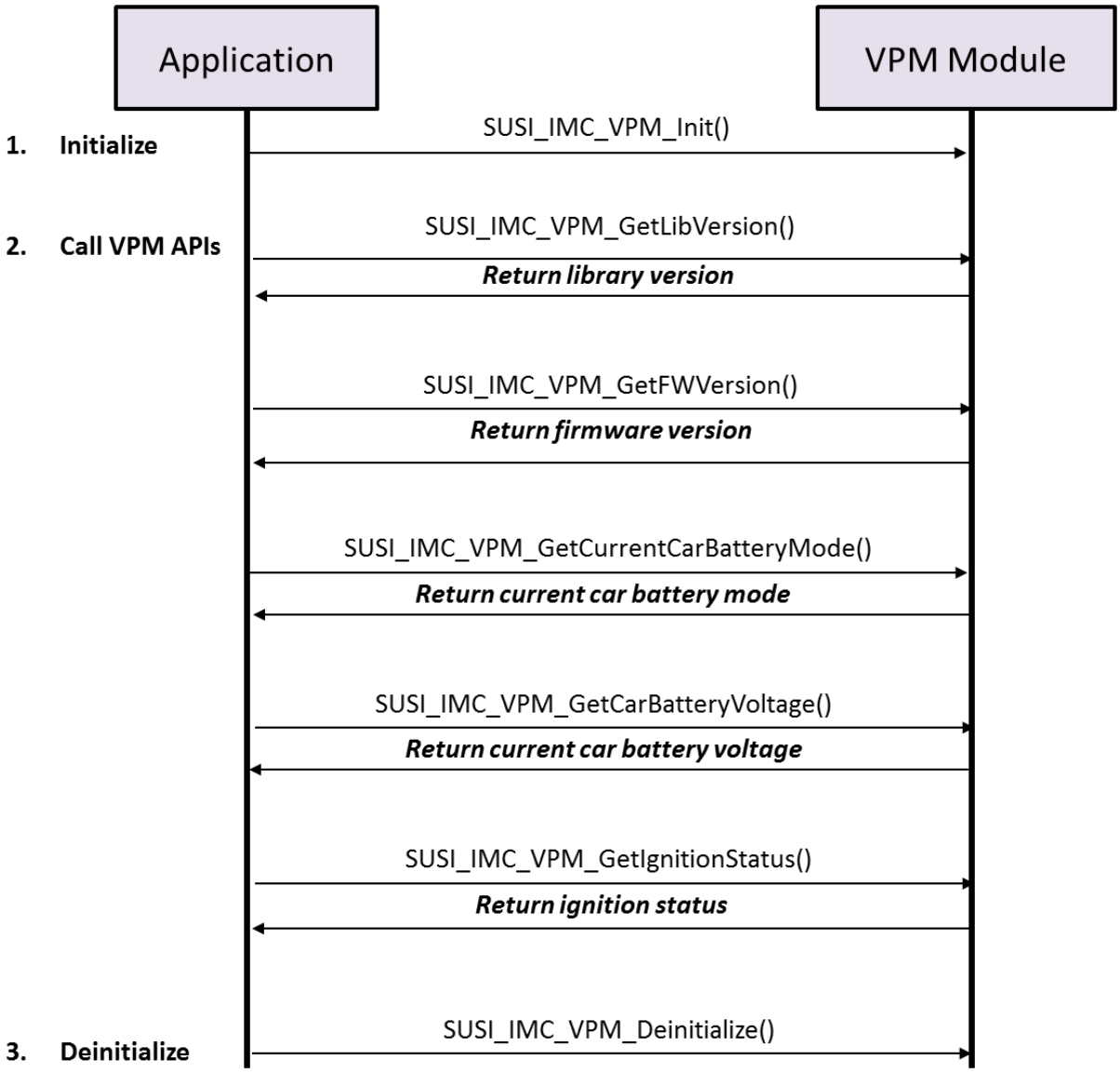
The class for VPM APIs for Android is **mrm.client.SUSIServiceClient**.

For the usage, please refer to the section SUSI SRV Usage Overview.

5.4.1.2 Get VPM Status

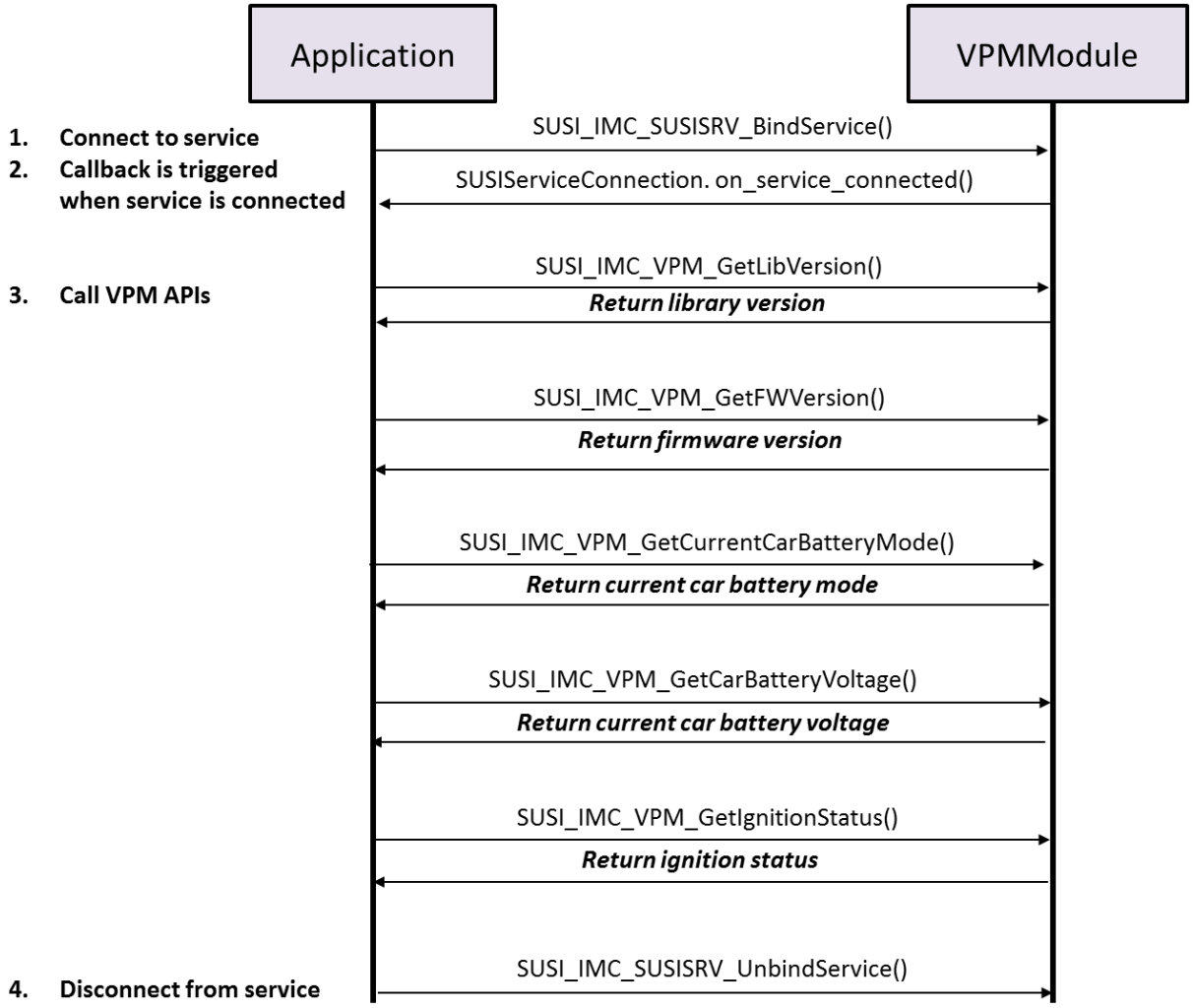
VPM API module provides APIs to get status data from VPM. The following section describe the usage of the APIs..

Windows/Linux



You can use SUSI_IMC_VPM_GetLibVersion() to get library version, SUSI_IMC_VPM_GetFWVersion() to check firmware version, SUSI_IMC_VPM_GetCurrentCarBatteryMode() to check the current battery mode, SUSI_IMC_VPM_GetCarBatteryVoltage() to check current car battery voltage and SUSI_IMC_VPM_GetIgnitionStatus() to check ignition status.

Android



You can use SUSI_IMC_VPM_GetLibVersion() to get library version, SUSI_IMC_VPM_GetFWVersion() to check firmware version, SUSI_IMC_VPM_GetCurrentCarBatteryMode() to check the current battery mode, SUSI_IMC_VPM_GetCarBatteryVoltage() to check current car battery voltage and SUSI_IMC_VPM_GetIgnitionStatus() to check ignition status.

5.4.1.3 Ignition Control

The VPM monitors the events of ignition ON/OFF and handles the power sequence of TREK.

- **Ignition ON Event**

When Ignition status is switched from OFF to ON, the VPM will count "Ignition ON delay" period to wait signal become stable. After count, the VPM will power on the TREK device.

If the ignition signal violating the conditions (e.g switched from ON to OFF) during the "Ignition on delay", the TREK will not be booted.

The "Ignition ON delay" can be adjust by using SUSI_IMC_VPM_SetIgnOnDelay() and can be get by using SUSI_IMC_VPM_GetIgnOffEventDelay() .

- **Ignition OFF Event**

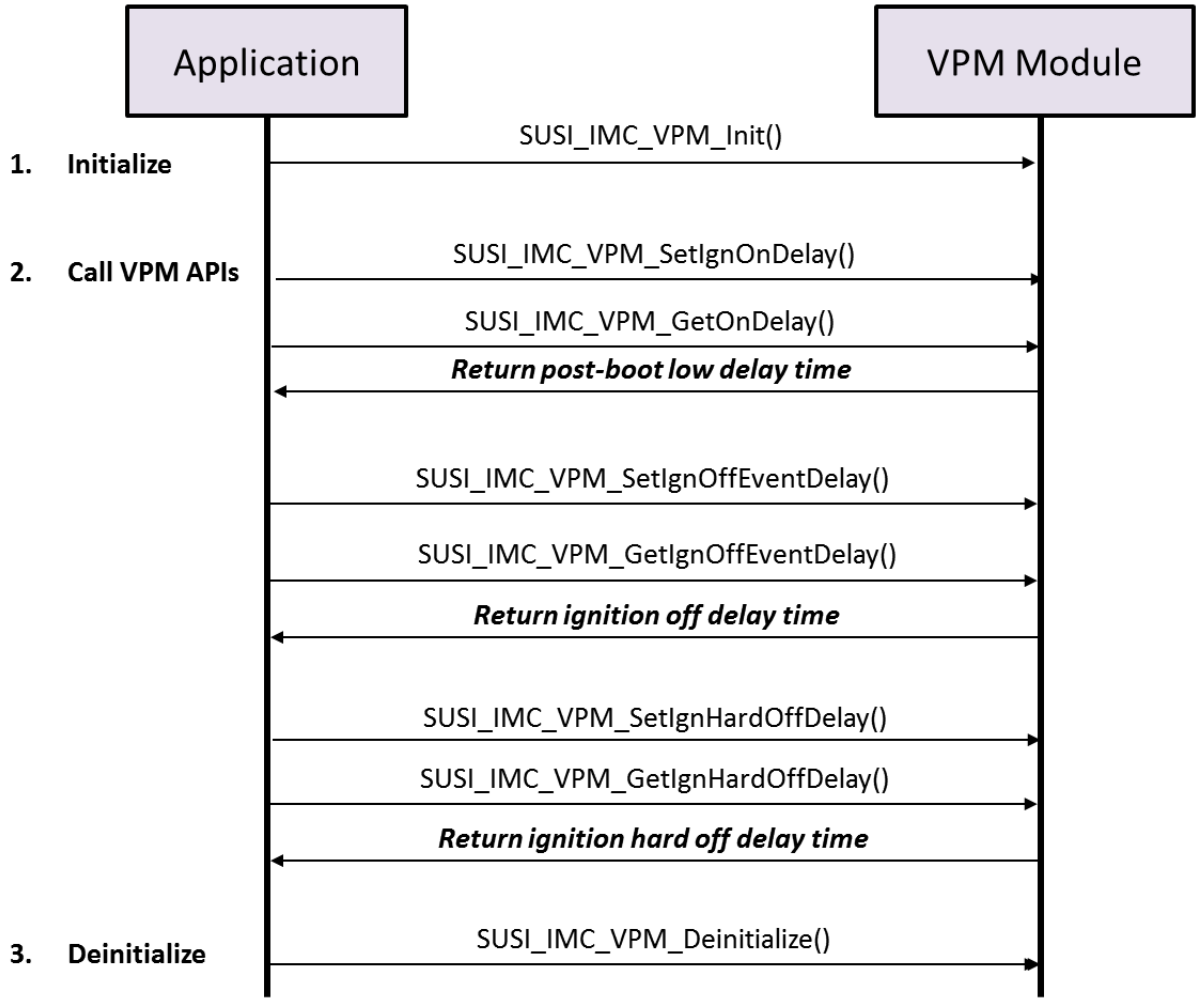
When Ignition status is switched from ON to OFF, the VPM will start counting "Ignition OFF delay". If the ignition signal violating the conditions (e.g switched from OFF to ON) during the period, the VPM will stop counting and stop the power off sequence.

After the "Ignition OFF delay" period expired, the VPM then start power off sequence and start counting "Hard OFF delay" and then power off the TREK device. Once the VPM start counting "Hard OFF delay", the count can not be canceled. The TREK device will be forced power off by the VPM when "Hard OFF delay" expired to avoid unexpected OS freeze.

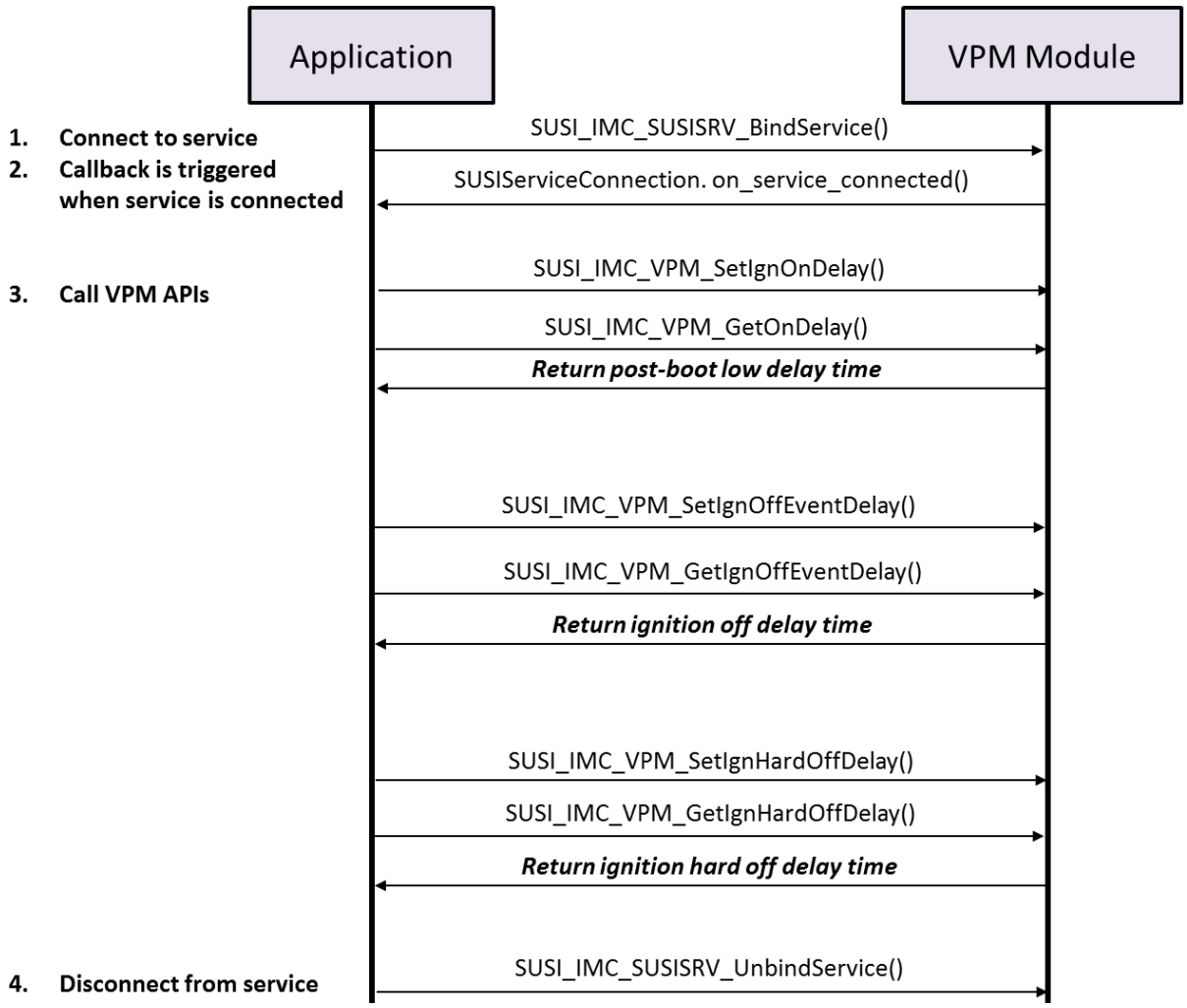
The "Hard OFF delay" can be adjust by using SUSI_IMC_VPM_SetIgnHardOffDelay() and can be get by using SUSI_IMC_VPM_GetIgnHardOffDelay() .

The following section describe the usage of the APIs.

Windows/Linux



Android


Application
VPM Module
1. Connect to service
2. Callback is triggered when service is connected
3. Call VPM APIs
4. Disconnect from service
SUSI_IMC_SUSISRV_BindService()
SUSIServiceConnection. on_service_connected()
SUSI_IMC_VPM_SetIgnOnDelay()
SUSI_IMC_VPM_GetOnDelay()
Return post-boot low delay time
SUSI_IMC_VPM_SetIgnOffEventDelay()
SUSI_IMC_VPM_GetIgnOffEventDelay()
Return ignition off delay time
SUSI_IMC_VPM_SetIgnHardOffDelay()
SUSI_IMC_VPM_GetIgnHardOffDelay()
Return ignition hard off delay time
SUSI_IMC_SUSISRV_UnbindService()

5.4.1.4 AT Mode / Keep Alive Mode

There are two mode settings for VPM to control how the TREK device should be booted up - the AT-Mode and the Keep-Alive-Mode :

- **Keep-Alive-Mode**

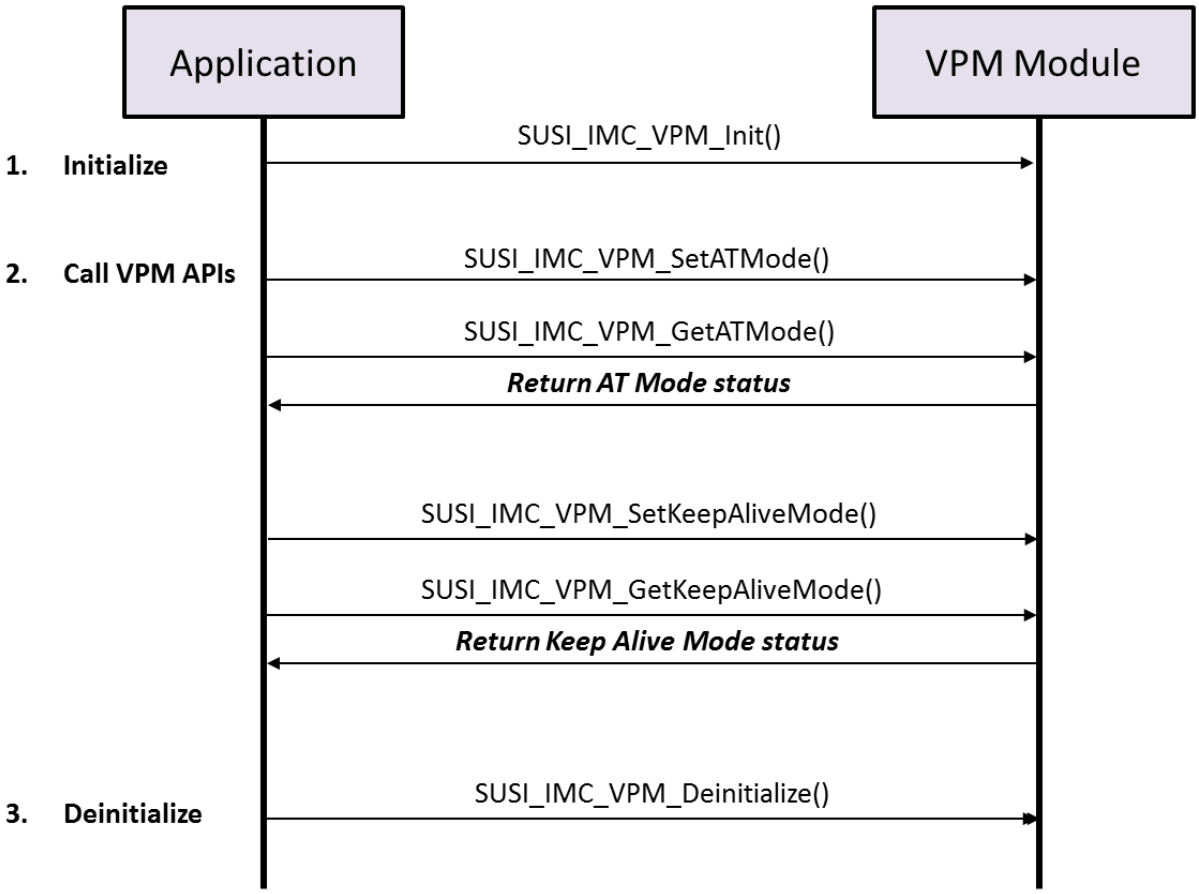
This mode decides whether TREK device should keeps power on.
The Keep-Alive-Mode setting can be enabled or disabled. If Keep-Alive-Mode is enabled, VPM will check whether the ignition status is ON when when the TREK device goes to power off or suspend status. If ignition status is ON, VPM will boot up TREK device immediately , otherwise the TREK device will not be booted up..
You can use SUSI_IMC_VPM_SetKeepAliveMode() to set status of Keep-Alive-Mode and use SUSI_IMC_VPM_GetKeepAliveMode() to get status of Keep-Alive-Mode.

- **AT-Mode**

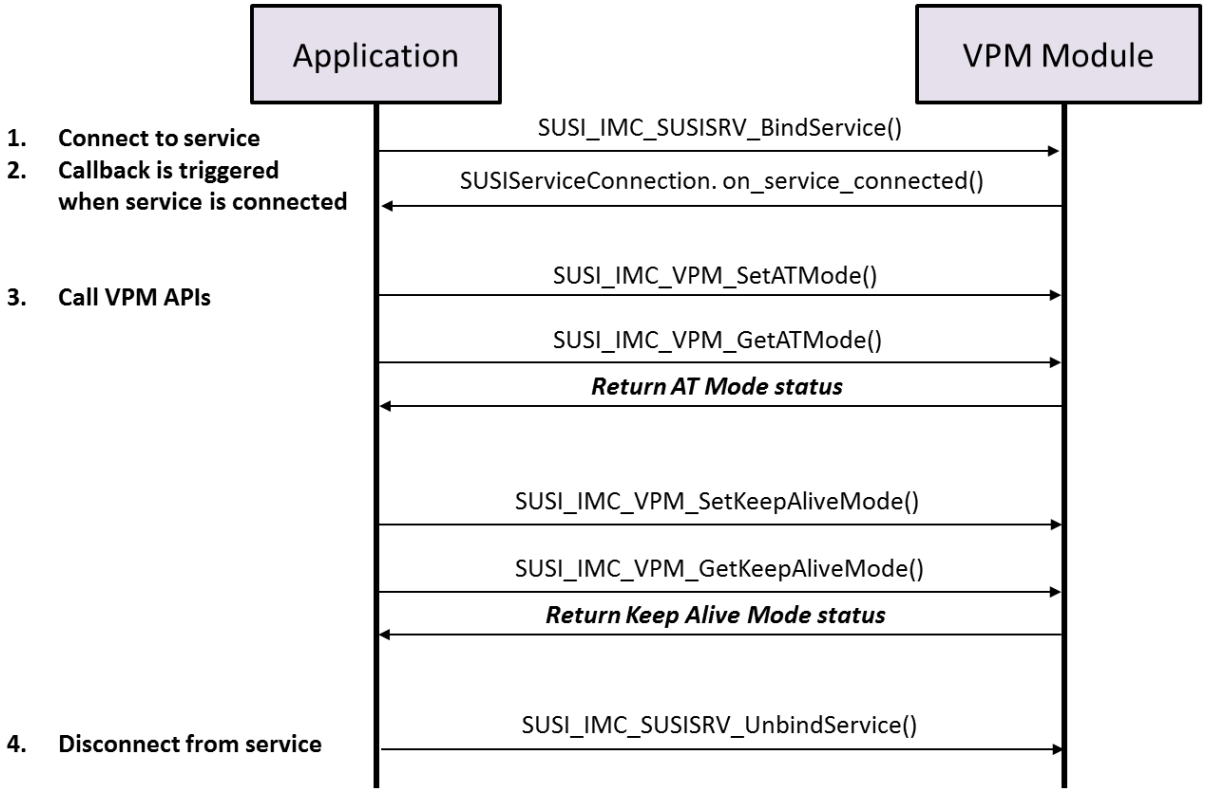
This mode decides whether TREK device should be booted up when the electric current is turned on.
The AT-Mode setting can be enabled or disabled. If AT-Mode is enabled, VPM will check whether the ignition status is ON when when the electric current is turned on. If ignition status is ON, VPM will start counting ignition ON delay then boot up TREK device, otherwise the TREK device will not be booted up..
For the definition of ignition on delay and the related sequence, please refer to ignition control section.
You can use SUSI_IMC_VPM_SetATMode() to set status of AT-Mode and use SUSI_IMC_VPM_GetATMode() to get status of AT-Mode.

If both mode are enabled, the Keep-Alive-Mode is prior than AT-Mode.
The following section describe the usage of the APIs.

Windows/Linux



Android

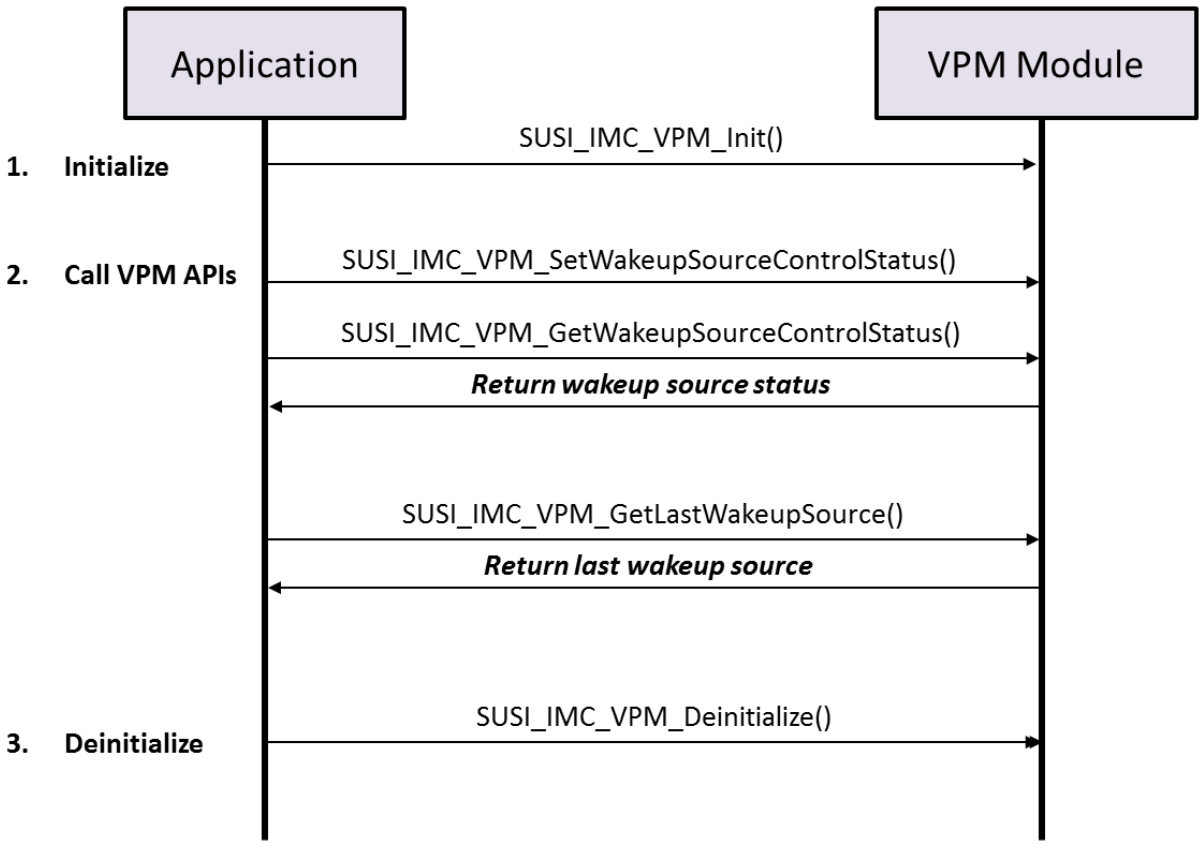

Application
VPM Module
1. Connect to service
2. Callback is triggered when service is connected
SUSI_IMC_SUSISRV_BindService()
SUSIServiceConnection. on_service_connected()
3. Call VPM APIs
SUSI_IMC_VPM_SetATMode()
SUSI_IMC_VPM_GetATMode()
Return AT Mode status
SUSI_IMC_VPM_SetKeepAliveMode()
SUSI_IMC_VPM_GetKeepAliveMode()
Return Keep Alive Mode status
4. Disconnect from service
SUSI_IMC_SUSISRV_UnbindService()

5.4.1.5 Wakeup Source

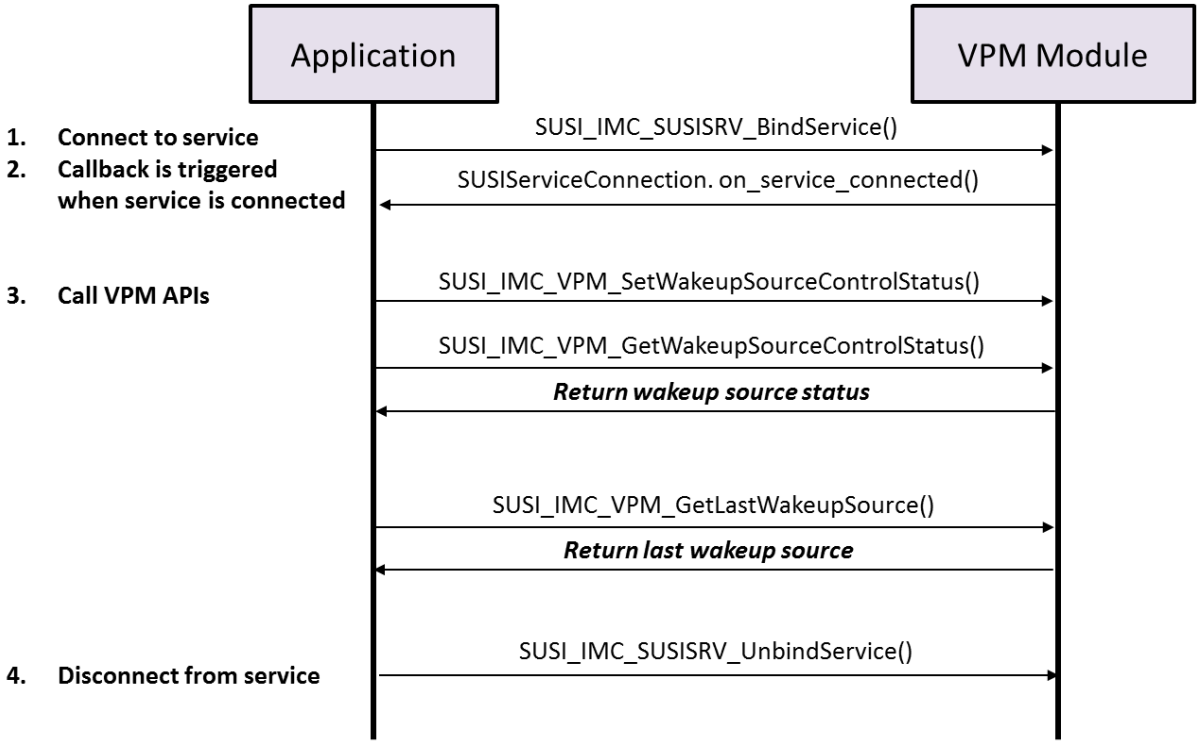
VPM allow user to set multiple device/conditions as wakeup sources of TREK device. If any of the enabled wakeup source is triggered, the TREK device will be booted up.
VPM support wakeup sources like G-Sensor, WWAN SMS, Alarm, Ignition, Power Button, Reset Button, DI pins and etc.

You can use SUSI_IMC_VPM_SetWakeupSourceControlStatus() to enable/disable wakeup source and use SUSI_IMC_VPM_GetWakeupSourceControlStatus() to get enable status of wakeup source.
The following section describe the usage of the APIs.

Windows/Linux



Android


Application
VPM Module
1. Connect to service
2. Callback is triggered when service is connected
3. Call VPM APIs
4. Disconnect from service
SUSI_IMC_SUSISRV_BindService()
SUSIServiceConnection. on_service_connected()
SUSI_IMC_VPM_SetWakeupSourceControlStatus()
SUSI_IMC_VPM_GetWakeupSourceControlStatus()
Return wakeup source status
SUSI_IMC_VPM_GetLastWakeupSource()
Return last wakeup source
SUSI_IMC_SUSISRV_UnbindService()

5.4.1.6 Alarm Wakeup

VPM provide alarm wakeup functions. You can set the alarm time to specify when VPM should wakeup(power on) the device.

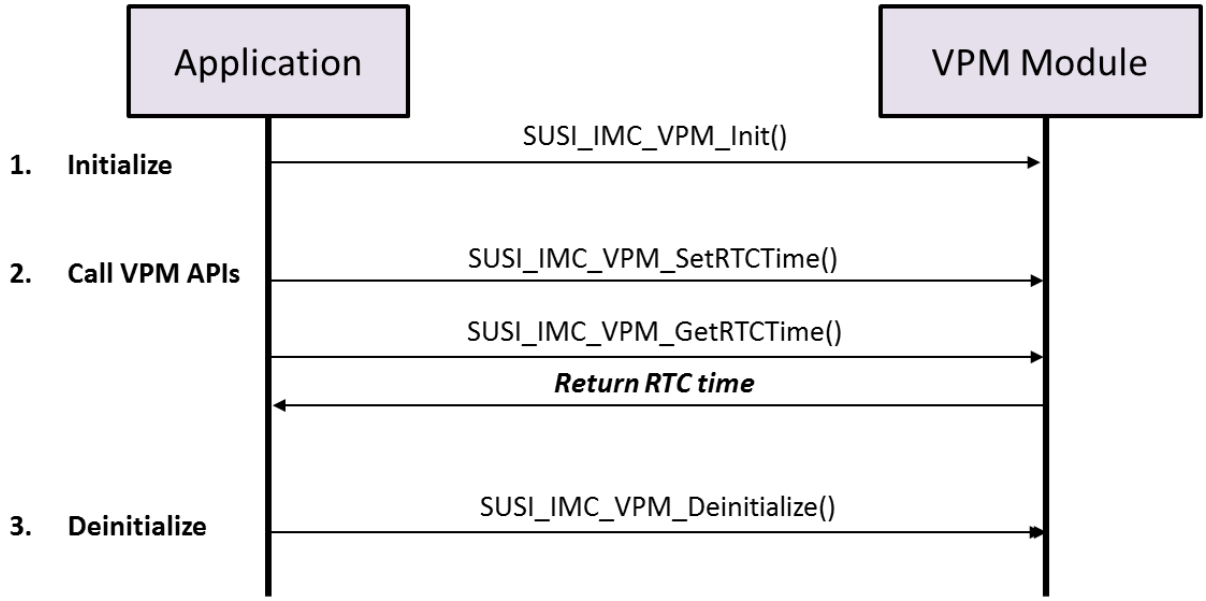
VPM holds an internal RTC. To set/get the RTC time, you can use SUSI_IMC_VPM_SetRTCTime() to set correct RTC time and call SUSI_IMC_VPM_GetRTCTime() .

To get/set the alarm time and specify how the alarm work, you can use SUSI_IMC_VPM_SetAlarmTime() and SUSI_IMC_VPM_SetAlarmTime(). You can set the alarm time and specify VPM to wakeup device HOURLY, DAILY or WEEKLY.

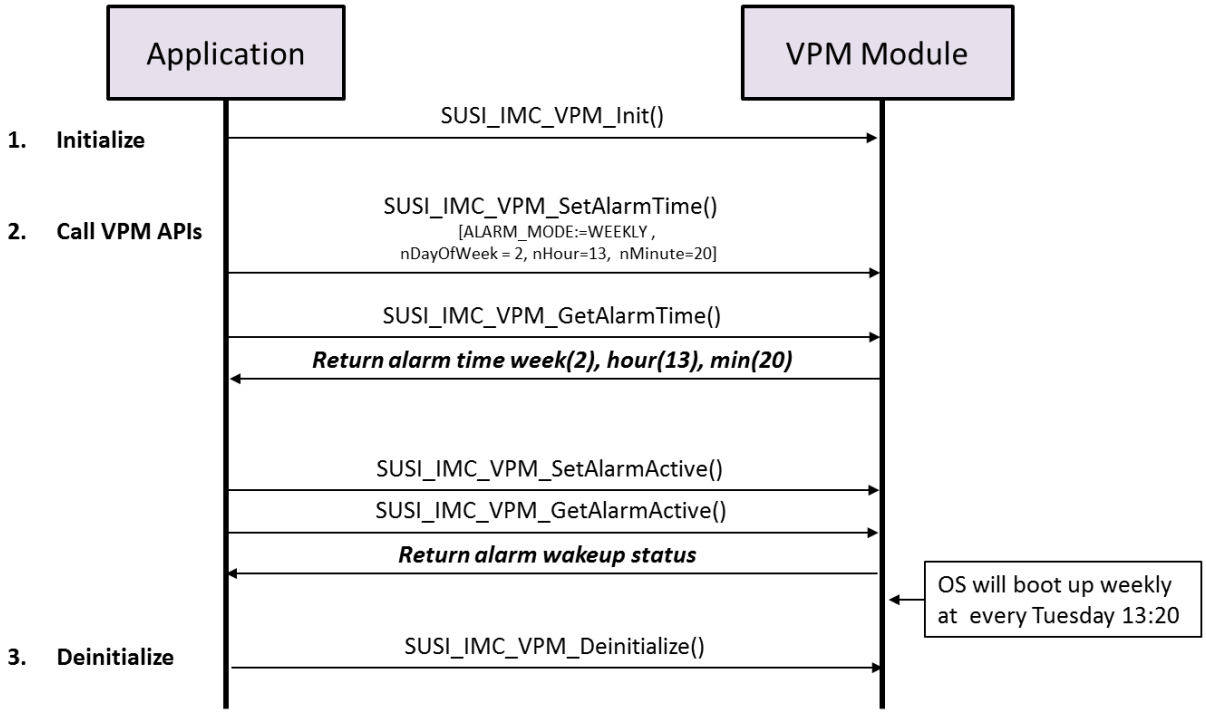
To enable/disable the alarm wakeup function, you can use SUSI_IMC_VPM_SetAlarmActive() . To get alarm wakeup status, use SUSI_IMC_VPM_GetAlarmActive().

Windows/Linux

Get/Set RTC:

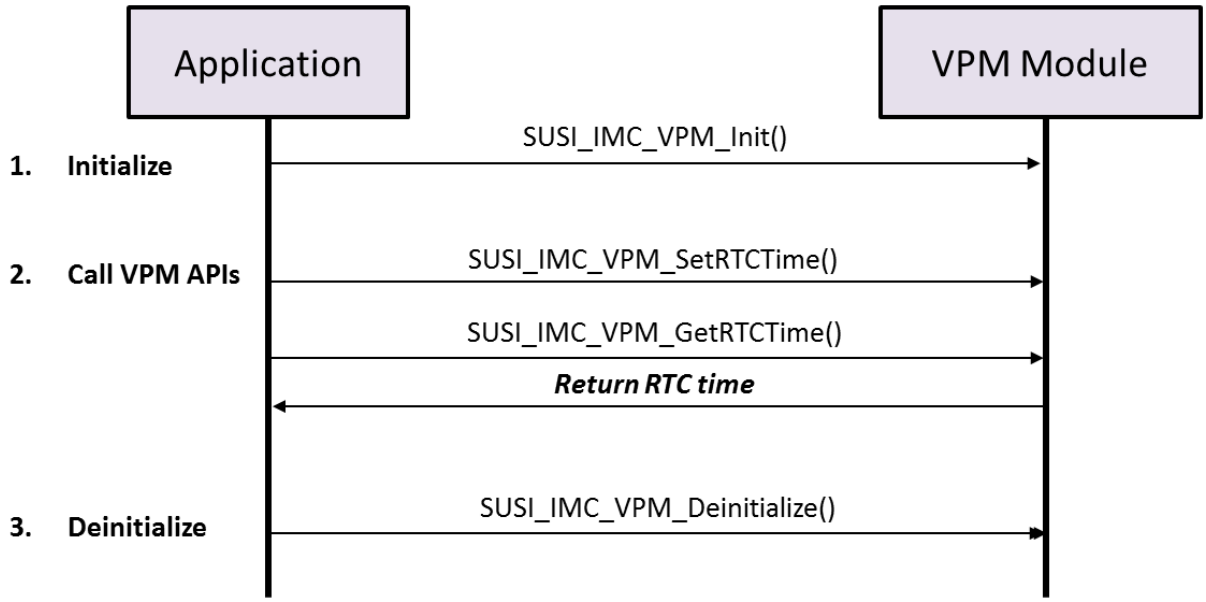


Get/Set Alarm:

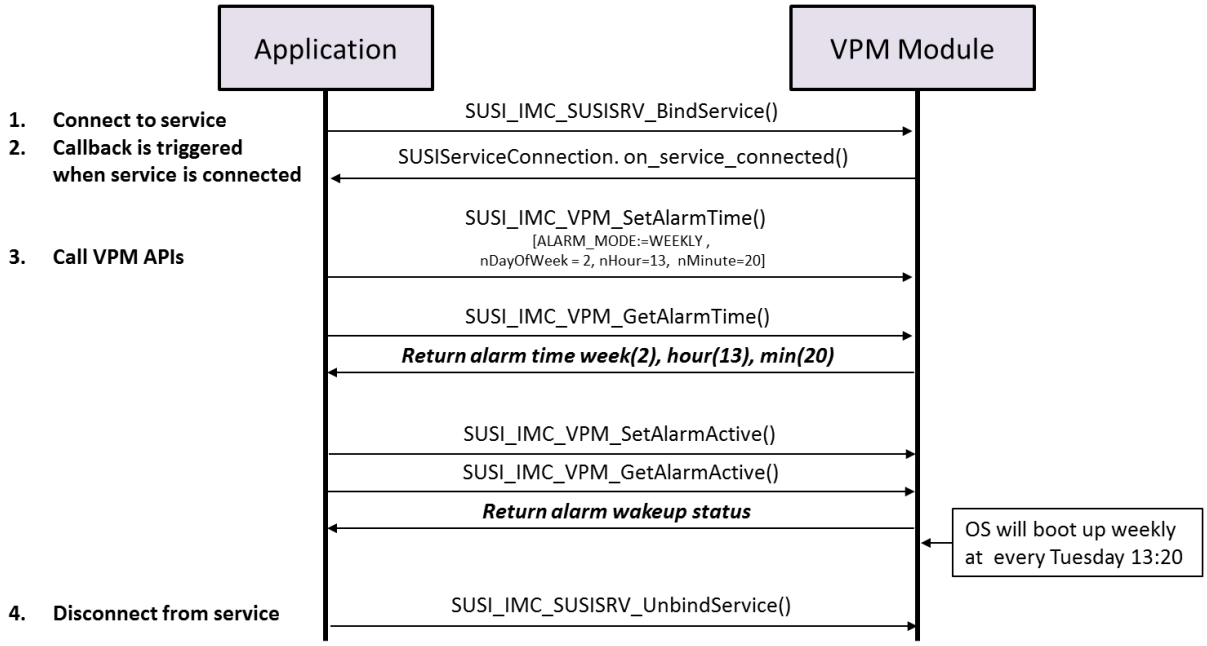

Application
VPM Module
1. Initialize
SUSI_IMC_VPM_Init()
2. Call VPM APIs
SUSI_IMC_VPM_SetAlarmTime()
[ALARM_MODE:=WEEKLY ,
nDayOfWeek = 2, nHour=13, nMinute=20]
SUSI_IMC_VPM_GetAlarmTime()
Return alarm time week(2), hour(13), min(20)
SUSI_IMC_VPM_SetAlarmActive()
SUSI_IMC_VPM_GetAlarmActive()
Return alarm wakeup status
OS will boot up weekly
at every Tuesday 13:20
3. Deinitialize
SUSI_IMC_VPM_Deinitialize()

Android

Get/Set RTC:



Get/Set Alarm:



5.4.1.7 Low Battery Protection

VPM provides low battery protection (LBP) functions which monitors whether the voltage of battery is above specified threshold and control the power of TREK device.
If LBP function is enabled, when the battery voltage is under the specified threshold (low battery), VPM will power off the TREK device for safety purpose.

The LBP functions includes Pre-boot LBP and Post-boot LBP:

- **Pre-boot LBP**

"Pre-boot" is defined as "before OS boot up". The pre-boot LBP function is able to do the battery voltage check before the OS boot up and decide whether to boot up the TREK device.
If Pre-boot LBP enabled, VPM check whether battery voltage is equal to or above the specified threshold for a specified "LBP low delay" period, if the battery voltage is stable, VPM allow OS to boot up. Otherwise, the OS will not boot up.

The VPM pre-boot LBP function supports 12V/24V battery system.
For setting/getting the pre-boot LBP voltage threshold for 12V battery system, you can use SUSI_IMC_VPM_SetPrebootThresholdFor12VSystem() and SUSI_IMC_VPM_GetPrebootLBPThresholdFor12VSystem() .
For setting/getting the pre-boot LBP voltage threshold for 24V battery system, you can use SUSI_IMC_VPM_SetPrebootThresholdFor24VSystem() and SUSI_IMC_VPM_GetPrebootLBPThresholdFor24VSystem() .

For setting/getting the LBP low delay, you can use SUSI_IMC_VPM_SetLBPLowDelay() and SUSI_IMC_VPM_GetLBPLowDelay() .

And uou can enable/disable pre-boot LBP function by using SUSI_IMC_VPM_SetPrebootLBPStatus() and get the status of pre-boot LBP function by using SUSI_IMC_VPM_GetPrebootLBPStatus().

- **Post-boot LBP**

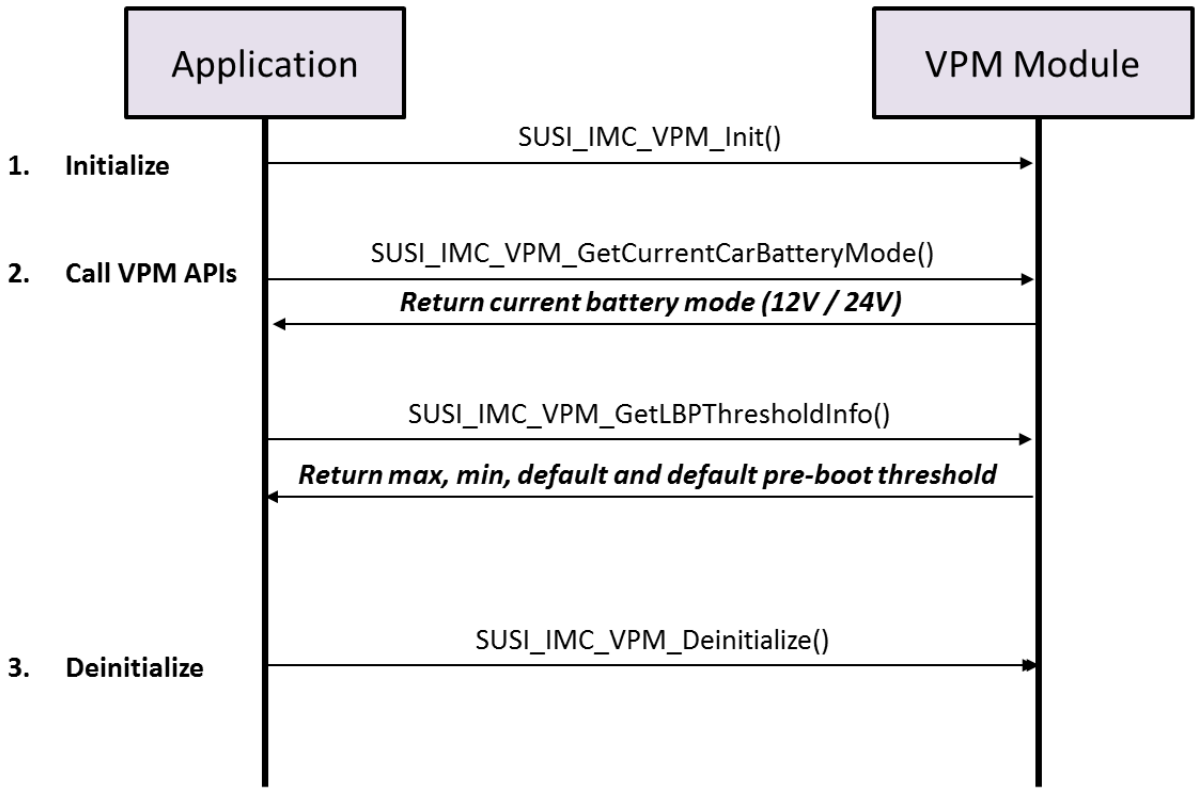
"Post-boot" is defined as "after OS boot up". The post-boot LBP function is able to do the battery voltage check after the OS boot up and decide whether to power off the TREK device when battery low. If Post-boot LBP enabled, VPM check whether battery voltage is under the specified threshold for a specified "LBP low delay" period, if the battery voltage is stable and low, VPM asks OS to shutdown and start counting "LBP low hard delay" period. If OS can not be shutdown during "LBP low hard delay" period, VPM will force power off TREK device.

The VPM post-boot LBP function supports 12V/24V battery system.
For setting/getting the post-boot LBP voltage threshold for 12V battery system, you can use SUSI_IMC_VPM_SetLBPThresholdFor12VSystem() and SUSI_IMC_VPM_GetLBPThresholdFor12VSystem() .
For setting/getting the post-boot LBP voltage threshold for 24V battery system, you can use SUSI_IMC_VPM_SetThresholdFor24VSystem() and SUSI_IMC_VPM_GetLBPThresholdFor24VSystem() .
For setting/getting the LBP low delay, you can use SUSI_IMC_VPM_SetLBPLowDelay() and SUSI_IMC_VPM_GetLBPLowDelay() .
For setting/getting the LBP low hard delay, you can use SUSI_IMC_VPM_SetLBPLowHardDelay() and SUSI_IMC_VPM_GetLBPLowHardDelay() .

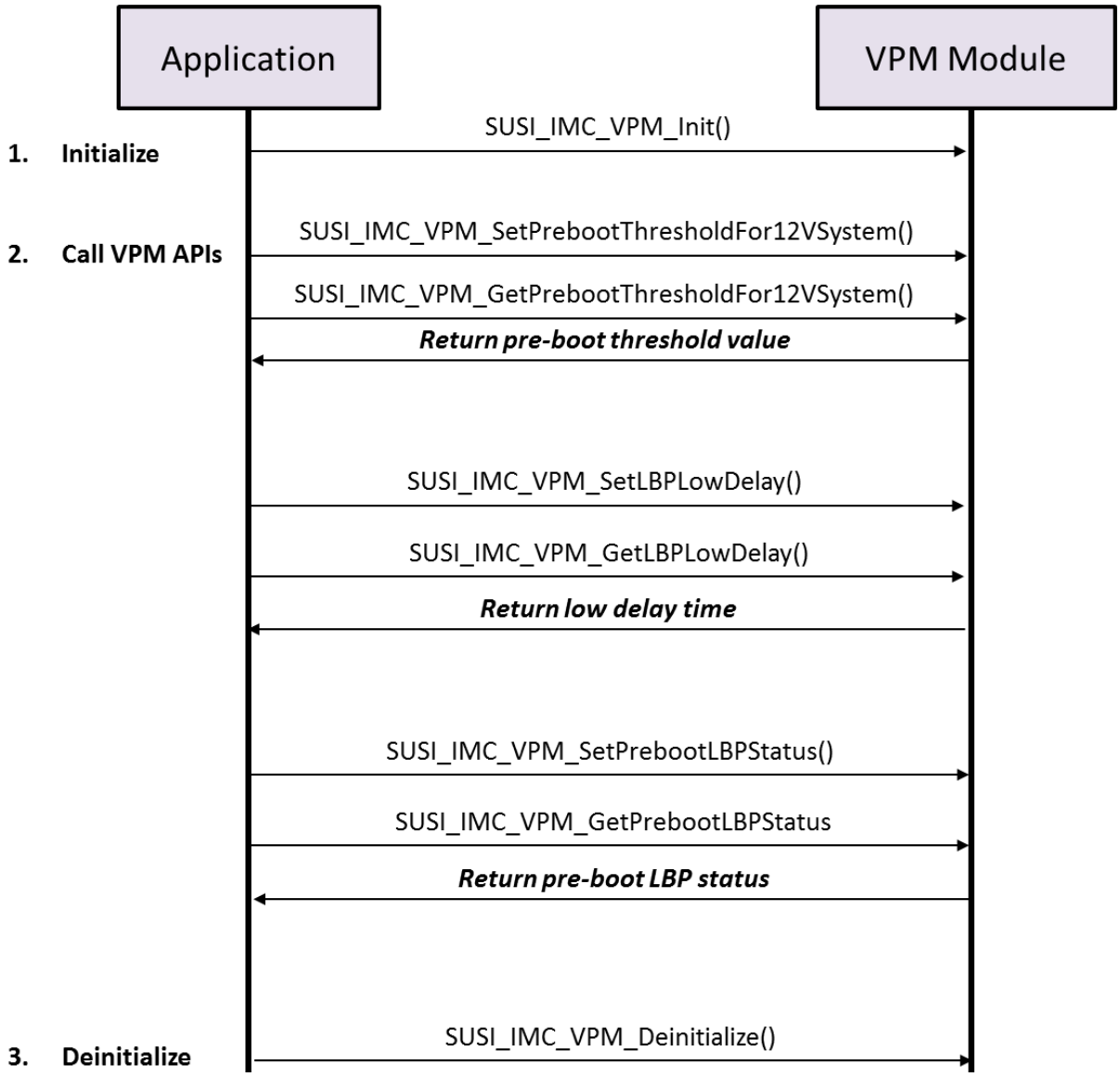
And you can enable/disable post-boot LBP function by using SUSI_IMC_VPM_SetBPStatus() and get the status of pre-boot LBP function by using SUSI_IMC_VPM_GetLBPStatus().

Windows/Linux

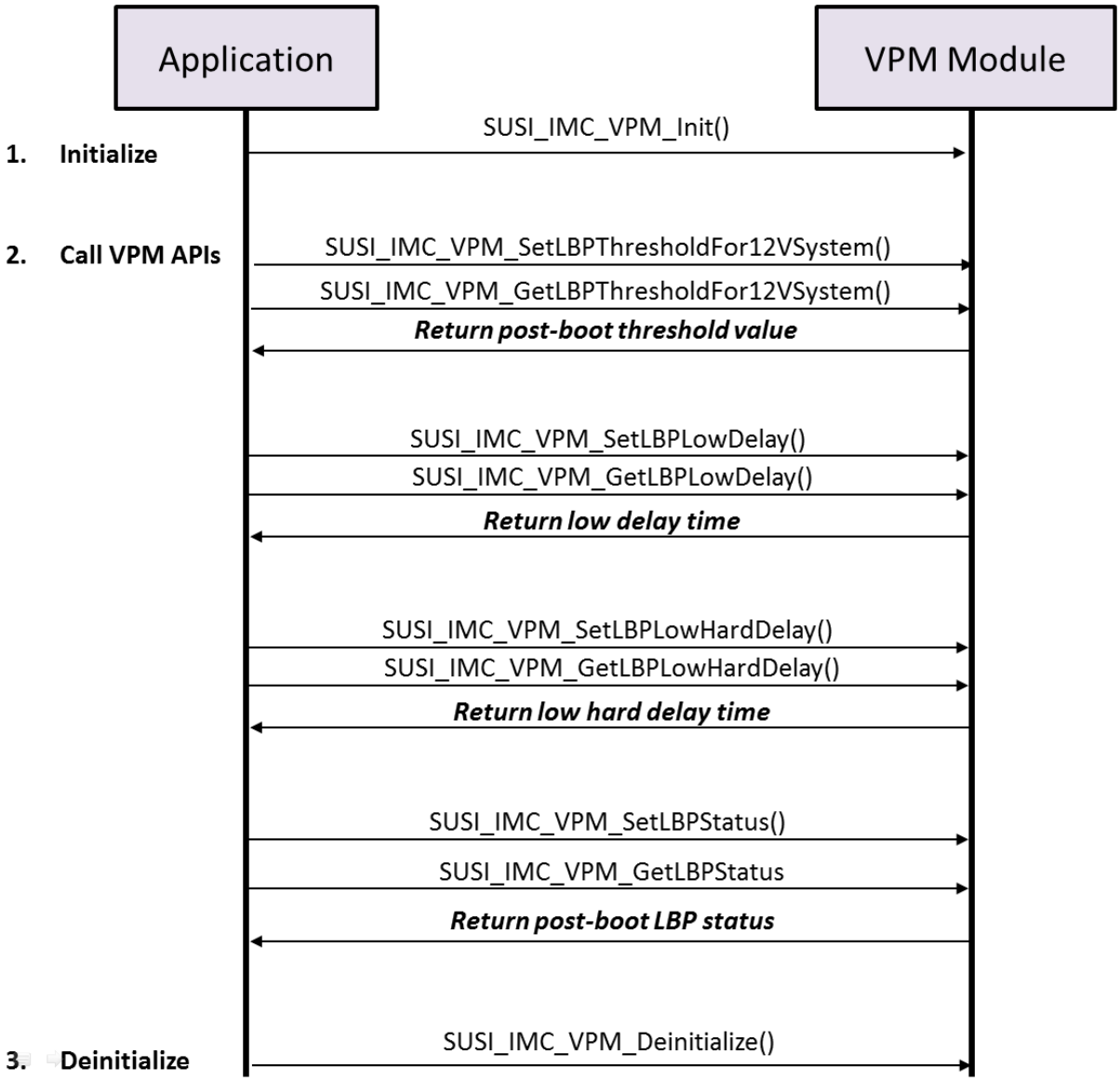
Get related information:



Set/Get Pre-boot LBP function:

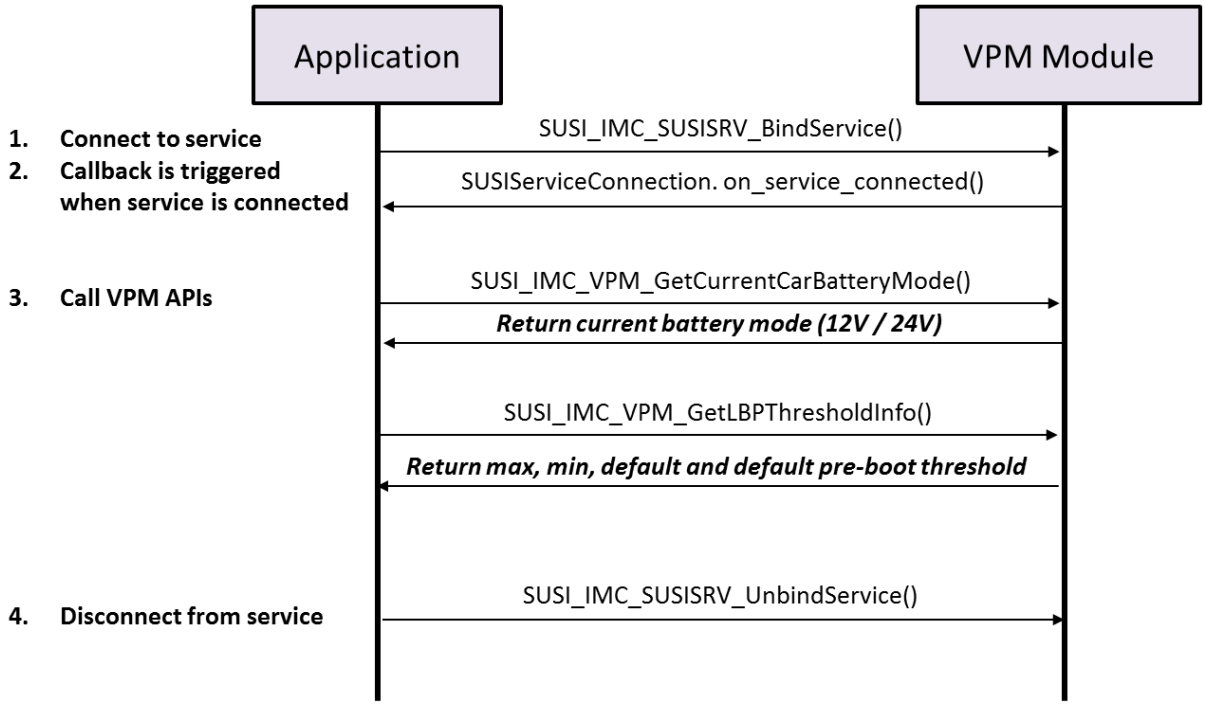


Set/Get Post-boot LBP function:

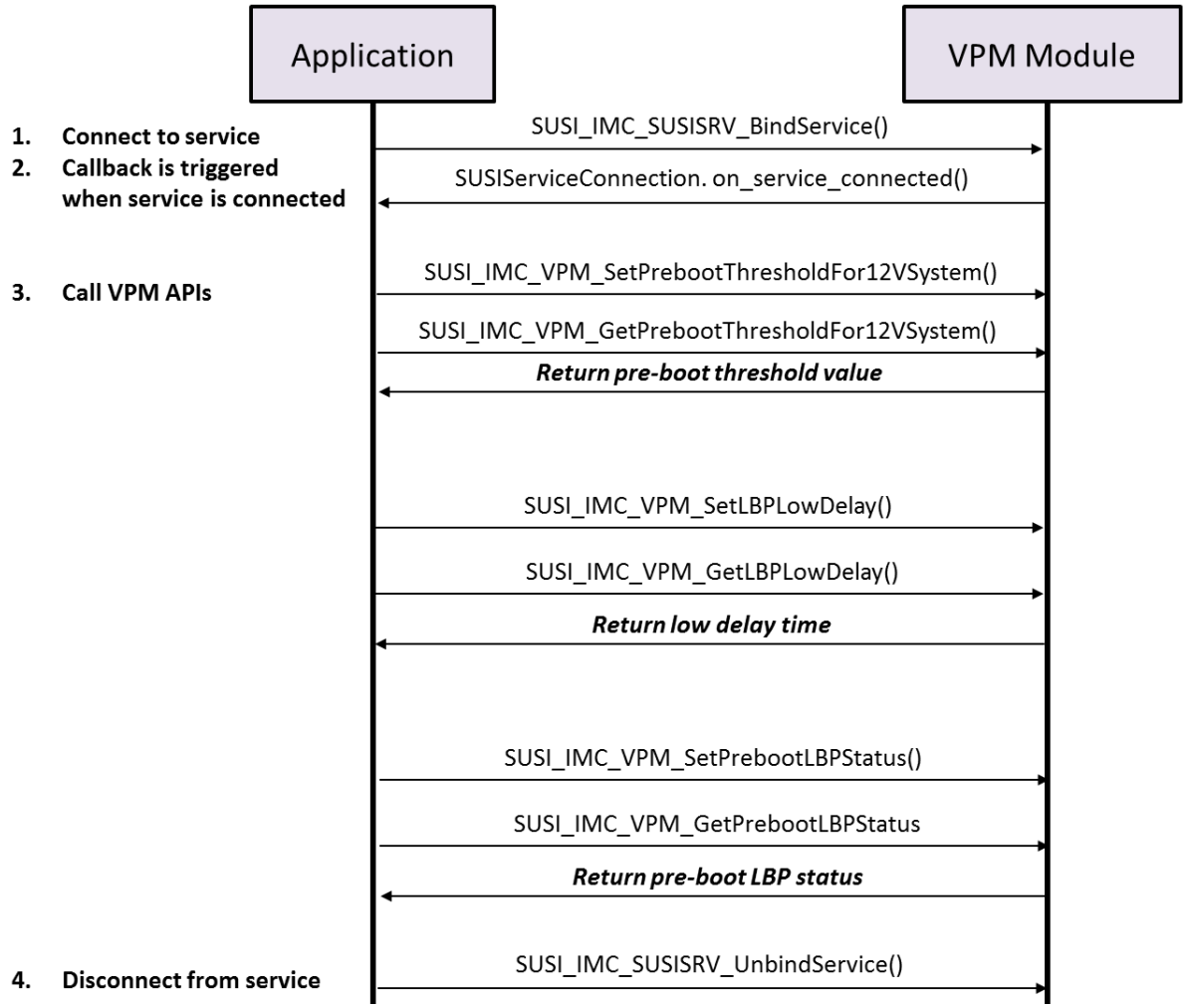

Application
VPM Module
1. Initialize
SUSI_IMC_VPM_Init()
2. Call VPM APIs
SUSI_IMC_VPM_SetLBPThresholdFor12VSystem()
SUSI_IMC_VPM_GetLBPThresholdFor12VSystem()
Return post-boot threshold value
SUSI_IMC_VPM_SetLBPLowDelay()
SUSI_IMC_VPM_GetLBPLowDelay()
Return low delay time
SUSI_IMC_VPM_SetLBPLowHardDelay()
SUSI_IMC_VPM_GetLBPLowHardDelay()
Return low hard delay time
SUSI_IMC_VPM_SetLBPStatus()
SUSI_IMC_VPM_GetLBPStatus
Return post-boot LBP status
3. Deinitialize
SUSI_IMC_VPM_Deinitialize()

Android

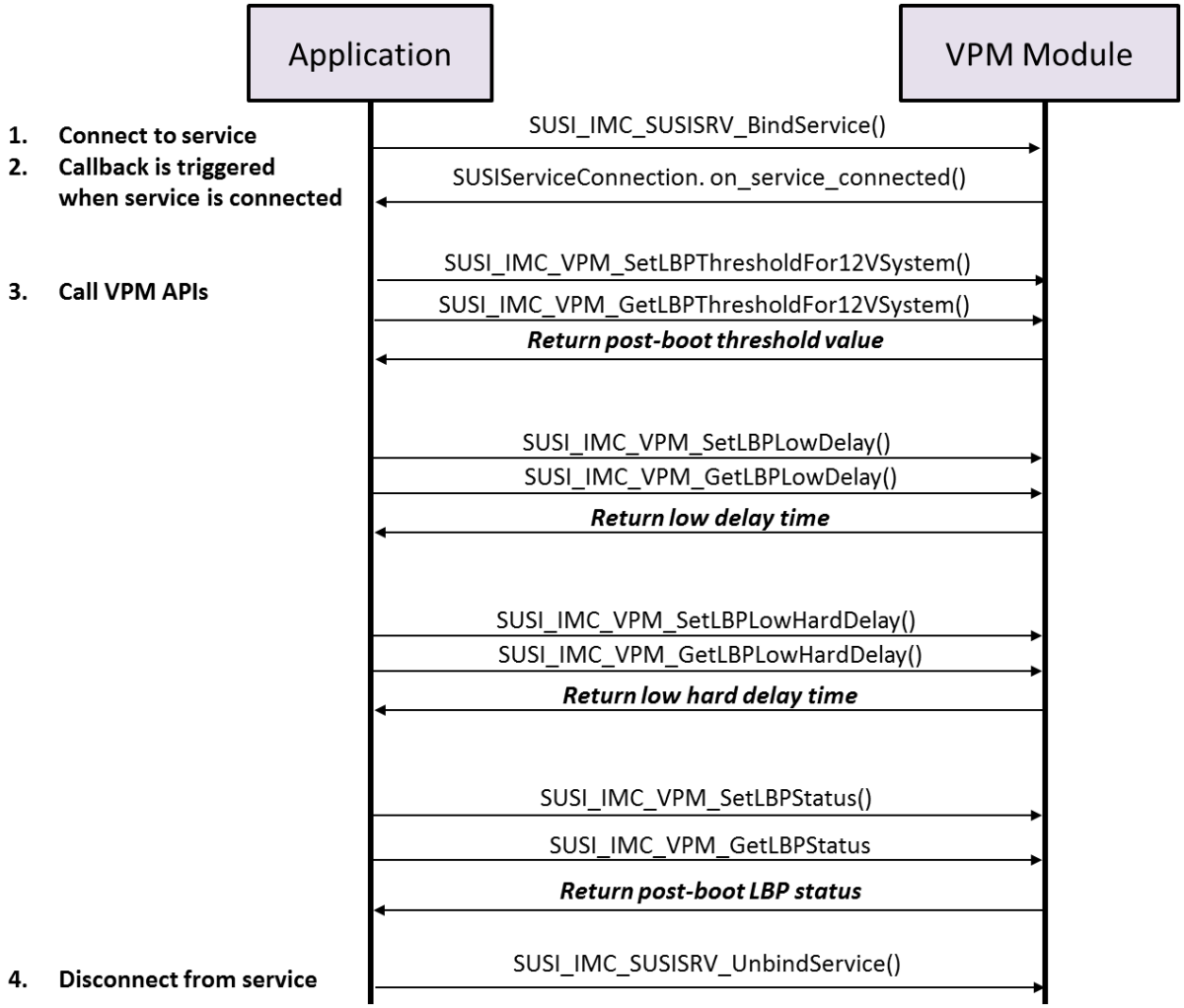
Get related information:



Set/Get Pre-boot LBP function:



Set/Get Post-boot LBP function:


Application
VPM Module
1. Connect to service
2. Callback is triggered when service is connected
3. Call VPM APIs
4. Disconnect from service
SUSI_IMC_SUSISRV_BindService()
SUSIServiceConnection. on_service_connected()
SUSI_IMC_VPM_SetLBPThresholdFor12VSystem()
SUSI_IMC_VPM_GetLBPThresholdFor12VSystem()
Return post-boot threshold value
SUSI_IMC_VPM_SetLBPLowDelay()
SUSI_IMC_VPM_GetLBPLowDelay()
Return low delay time
SUSI_IMC_VPM_SetLBPLowHardDelay()
SUSI_IMC_VPM_GetLBPLowHardDelay()
Return low hard delay time
SUSI_IMC_VPM_SetLBPStatus()
SUSI_IMC_VPM_GetLBPStatus
Return post-boot LBP status
SUSI_IMC_SUSISRV_UnbindService()

5.4.2 Enumeration

5.4.2.1 Windows/Linux

ALARM_MODE enum

- (-1) **NO_ALARM** - The computer is not enable alarm wakeup.
- (0) **HOURLY** - The computer is wakeup hourly by setting alarm minute.
- (1) **DAILY** - The computer is wakeup daily by setting alarm hour and minute.
- (2) **WEEKLY** - The computer is wakeup weekly by setting alarm day of week, hour and minute.

WAKEUP_SOURCE enum

- (-1) **WAKEUP_SOURCE_NONE** - The wakeup source is not set.
- (0) **WAKEUP_SOURCE_POWER_BUTTON** - Computer wakeup power button pushed.
- (1) **WAKEUP_SOURCE_IGNITION** - Computer wakeup form ignition off to on.
- (2) **WAKEUP_SOURCE_WWAN** - Computer wakeup form WWAN module.
- (3) **WAKEUP_SOURCE_GSENSOR** - Computer wakeup form gsensor.
- (4) **WAKEUP_SOURCE_D0** - Computer wakeup form digital input 0.
- (5) **WAKEUP_SOURCE_D1** - Computer wakeup form digital input 1.
- (6) **WAKEUP_SOURCE_ALARM** - Computer wakeup form RTC alarm.
- (7) **WAKEUP_SOURCE_HOT_KEY** - Computer wakeup form hotkey board.
- (8) **WAKEUP_SOURCE_D2** - Computer wakeup form digital input 2.
- (9) **WAKEUP_SOURCE_D3** - Computer wakeup form digital input 3.
- (10) **WAKEUP_SOURCE_RESET** - Computer wakeup form software reset or reset button.

SHUTDOWN_SOURCE enum

- (0) **SHUTDOWN_SOURCE_POWER_BUTTON** - Computer wakeup power button pushed.
- (1) **SHUTDOWN_SOURCE_IGNITION** - Computer wakeup form ignition off to on.

IGNITION_MODE enum

- (0) **IGNITION_NONE** - The ignition mod is not set.
- (1) **IGNITION_POWEROFF** - The ignition mod on power off.
- (2) **IGNITION_SUSPEND** - The ignition mod on suspend.
- (3) **IGNITION_SIZE** - The ignition mod size.

5.4.2.2 Android

ALARM_MODE enum

class:

mrm.define.IMC_ENUM.ALARM_MODE

Enum:

Name	Type	value	Comment
HOURLY	int	0	The computer is wakeup hourly by setting alarm minute.
DAILY	int	1	The computer is wakeup daily by setting alarm hour and minute.
WEEKLY	int	2	The computer is wakeup weekly by setting alarm day of week, hour and minute.

Remark:

Use the method **getValue()** to get the enum value.

ex: MRM_ENUM.ALARM_MODE.DAILY**.getValue()** returns 1.

Please refer to sample code for detailed usage.

WAKEUP_SOURCE enum

class:

mrm.define.IMC_ENUM.**WAKEUP_SOURCE**

Enum:

Name	Type	value	Comment
WAKEUP_SOURCE_POWER_BUTTON	int	0	Computer wakeup from power button pushed.
WAKEUP_SOURCE_IGNITION	int	1	Computer wakeup from ignition off to on.
WAKEUP_SOURCE_WWAN	int	2	Computer wakeup from WWAN module.
WAKEUP_SOURCE_GSENSOR	int	3	Computer wakeup from G-sensor
WAKEUP_SOURCE_D0	int	4	Computer wakeup from digital input 0.
WAKEUP_SOURCE_D1	int	5	Computer wakeup fromfrom digital input 1.
WAKEUP_SOURCE_ALARM	int	6	Computer wakeup form RTC alarm.
WAKEUP_SOURCE_HOT_KEY	int	7	Computer wakeup from hotkey board.
WAKEUP_SOURCE_D2	int	8	Computer wakeup from digital input 2.
WAKEUP_SOURCE_D3	int	9	Computer wakeup from digital input 3.
WAKEUP_SOURCE_RESET	int	10	Computer wakeup from software reset or reset button.

Remark:

Use the method **getValue()** to get the enum value.

ex: MRM_ENUM.WAKEUP_SOURCE.WAKEUP_SOURCE_IGNITION**.getValue()** returns 1.

Please refer to sample code for detailed usage.

IGNITION_MODE enum

class:

mrm.define.IMC_ENUM.IGNITION_MODE

Enum:

Name	Type	value	Comment
NONE	int	0	The ignition mode is not set.
POWEROFF	int	1	The ignition off to power off.
SUSPEND	int	2	The ignition off to suspend.

Remark:

Use the method **getValue()** to get the enum value.

ex: MRM_ENUM.IGNITION_MODE.POWEROFF**.getValue()** returns 1.

Please refer to sample code for detailed usage.

CAR_BATTERY_MODE enum

class:

mrm.define.IMC_ENUM.CAR_BATTERY_MODE

Enum:

Name	Type	value	Comment
CAR_BATTERY_UNKNOWN	int	-1	
CAR_BATTERY_DEFAULT	int	0	
CAR_BATTERY_48V	int	1	
CAR_BATTERY_24V	int	2	
CAR_BATTERY_12V	int	3	

Remark:

Use the method **getValue()** to get the enum value.

ex: MRM_ENUM.CAR_BATTERY_MODE.CAR_BATTERY_48V**.getValue()** returns 1.

Please refer to sample code for detailed usage.

5.4.3 Constants

5.4.3.1 Android

Class:
mrm.define.**IMC_CONSTANT**

Fields:

Field Name	Type	Value
VPM_MAXIMUM_FIRMWARE_VERSION_LENGTH	int	32

5.4.4 Structure/Classes

5.4.4.1 Windows/Linux

VPM_RTC_TIME Structure

Syntax:

```
struct VPM_RTC_TIME
{
        unsigned char nYear;
        unsigned char nMonth;
        unsigned char nDay;
        unsigned char nDayOfWeek;
        unsigned char nHour;
        unsigned char nMinute;
        unsigned char nSecond;
};
typedef VPM_RTC_TIME* PVPM_RTC_TIME;
```

Description:

This structure defines the RTC Time.

Members:

nYear

Year. The range of the value is 0~99.

0 = 2000

99 = 2099

nMonth

Month. The range of the value is 1~12

nDay

Day. The range of the value is 1~31 depned on Month

nDayOfWeek

Day of week. The range of the value is 0 ~ 6.

0 = Sunday

nHour

Hour. The range of the value is 0 ~ 23.

nMinute

Minute. The range of the value is 0 ~ 59.

nSecond

Second. The range of the value is 0 ~ 59.

VPM_ALARM_TIME Structure

Syntax:

```
struct VPM_ALARM_TIME
{
        unsigned char nDayOfWeek;
            unsigned char nHour;
            unsigned char nMinute;
};
typedef VPM_ALARM_TIME* PVPM_ALARM_TIME;
```

Description:

This structure defines the alarm time .

Members:

nDayOfWeek

Day of week, The range of the value is 0 ~ 6. 0 = Sunday

nHour

Hour. The range of the value is 0 ~ 23.

nMinute

Minute. The range of the value is 0 ~ 59.

BACKUP_BATTERY_FLAG_INFO Structure

Syntax:

```
struct BACKUP_BATTERY_FLAG_INFO
{
        bool DSG;
        bool SOCF;
        bool SOC1;
        bool OCVTAKEN;
        bool CHG;
        bool FC;
        bool XCHG;
        bool CHG_INH;
        bool OTD;
        bool OTC;
};
typedef BACKUP_BATTERY_FLAG_INFO* PBACKUP_BATTERY_FLAG_INFO;
```

Description:

This structure defines the backup battery info .

Members:

DSG

Discharging detected.

SOCF

State-of-Charge-Threshold Final (SOCF Set %) reached.

SOC1

State-of-Charge-Threshold 1 (SOC1 Set) reached.

OCVTAKEN

Cleared on entry to relax mode and set to 1 when OCV measurement is performed in relax.

CHG

Charging allowed.

FC

Full-charged is detected.

XCHG

Charge Suspend Alert indicates the temperature is outside the range.

CHG_INH

Charge Inhibit indicates the temperature is outside the range.

OTD

Over-Temperature in Discharge condition is detected.

OTC

Over-Temperature in Charge condition is detected.

PBACKUP_BATTERY_INFO Structure

Syntax:

```
struct BACKUP_BATTERY_INFO
{
        int iMaxCapacity;
            int iCycleCount;
            BACKUP_BATTERY_FLAG_INFO BattryFlag;
};
typedef BACKUP_BATTERY_INFO* PBACKUP_BATTERY_INFO;
```

Description:

This structure defines the backup battery info .

Members:

iMaxCapacity

The backup battery max capacity.

iCycleCount

The backup battery cycle count.

5.4.4.2 Android

VPM_RTC_TIME

class:

mrm.define.vpm.**VPM_RTC_TIME**

Fields:

Field Name	Type	Input/Output	Comment
nYear	byte	in, out	Year. The range of the value is 0~99. 0 is for 2000, 99 is for 2099
nMonth	byte	in, out	Month. The range of the value is 1~12
nDay	byte	in, out	Day. The range of the value is 1~31 depends on Month
nDayOfWeek	byte	in, out	Day of week. The range of the value is 0 ~ 6. 0 is for Sunday
nHour	byte	in, out	Hour. The range of the value is 0 ~ 23.
nMinute	byte	in, out	Minute. The range of the value is 0 ~ 59.
nSecond	byte	in, out	Second. The range of the value is 0 ~ 59.

VPM_ALARM_TIME

class:

mrm.define.vpm.**VPM_ALARM_TIME**

Fields:

Field Name	Type	Input/Output	Comment
nDayOf Week	byte	in, out	Day of week. The range of the value is 0 ~ 6. 0 is for Sunday
nHour	byte	in, out	Hour. The range of the value is 0 ~ 23.
nMinute	byte	in, out	Minute. The range of the value is 0 ~ 59.

BACKUP_BATTERY_FLAG_INFO

class:

mrm.define.vpm.**VPM_RTC_TIME**

Fields:

Field Name	Type	Input/Output	Comment
DSG	boolean	out	Discharging detected.
SOCF	boolean	out	State-of-Charge-Threshold Final (SOCF Set %) reached.
SOC1	boolean	out	State-of-Charge-Threshold 1 (SOC1 Set) reached.
OCVTAKEN	boolean	out	Cleared on entry to relax mode and set to 1 when OCV measurement is performed in relax.
CHG	boolean	out	Charging allowed.
FC	boolean	out	Full-charged is detected.
XCHG	boolean	out	Charge Suspend Alert indicates the temperature is outside the range.
CHG_INH	boolean	out	Charge Inhibit indicates the temperature is outside the range.
OTD	boolean	out	Over-Temperature in Discharge condition is detected.
OTC	boolean	out	Over-Temperature in Charge condition is detected.

BACKUP_BATTERY_INFO

class:

mrm.define.vpm.**BACKUP_BATTERY_INFO**

Fields:

Field Name	Type	Input/Output	Comment
maxCapacity	int	out	The backup battery max capacity.
cycleCount	int	out	The backup battery cycle count.
backupBatteryFlagInfo	BACKUP_BATTERY_FLAG_INFO	out	Class defines the backup battery info.

BATTERY_INFO

class:

mrm.define.vpm.**BATTERY_INFO**

Fields:

Field Name	Type	Input/Output	Comment
batteryMaxVoltage	float	out	The max value of LBP threshold.
batteryMinVoltage	float	out	The minimum value of LBP threshold.
batteryDefVoltage	float	out	The default value of post-boot LBP threshold
batteryDefPrebootVoltage	float	out	The default value of pre-boot LBP threshold

5.4.5 APIs

5.4.5.1 SUSI_IMC_VPM_Initialize

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_Initialize**(IN BYTE port_number);

Description:

Initialize the VPM library.

Parameters:

port_number [in]

Specifies the port number of VPM.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

- It shown the UART mapping table as following:

Hardware Platform	OS Platform	Port ID/PATH
TREK-520	WinCE	6
TREK-520	Linux	/proc/vpm_serial
TREK-674 TREK-572	Windows	1
TREK-674 TREK-572	Linux	/dev/ttyS0
TREK-570	Windows	4
TREK-570	Linux	/dev/ttyS3

Note: Linux ignore the port_number argument and using the variable VPM_PATH which defined in /etc/susi/susi_config

5.4.5.2 SUSI_IMC_VPM_Deinitialize

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_Deinitialize**(void);

Description:

Deinitialize the VPM library.

Parameters:

None

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.4.5.3 SUSI_IMC_VPM_GetFWVersion

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_GetFWVersion**(OUT PPWM_FIRMWARE_INFO pVersion);
Android	short **SUSI_IMC_VPM_GetFWVersion**(byte[] pVersion)

Description:

Get the firmware version.

Parameters:

pversion [out]

Linux / Windows

Points to a struct that specifies the firmware information.

Android:

An allocated byte array of size VPM_MAXIMUM_FIRMWARE_VERSION_LENGTH large for storing the ASCII codes of firmware version string characters.

You can get the firmware version string by convert the byte array to String.

Please refer to the sample code for detailed usage.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.4.5.4 SUSI_IMC_VPM_GetLibVersion

Syntax:

CE / Windows	USHORT **SUSI_IMC_VPM_GetLibVersion**(OUT BYTE version[18]);

Description:

Get the library version from TREK SDK.

Parameters:

pVersion [out]

Point to read buffer that specifies the Power Management library's current version.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 CE

TREK-570 Windows

TREK-572 Windows

TREK-674 Windows

Remark:

The version buffer size must be greater than 18 bytes.

5.4.5.5 SUSI_IMC_VPM_GetPlatformName

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_GetPlatformName**(OUT PCHAR pName);
Android	short **SUSI_IMC_VPM_GetPlatformName**(byte[] pName)

Description:

Get the platform name.

Parameters:

pName [out]

Linux / Windows

Points to a string which end of '\0'.

Android:

An allocated byte array of size VPM_MAXIMUM_PLATFORM_NAME_LENGTH large for storing the ASCII codes of firmware version string characters.

You can get the platform name string by convert the byte array to String.

Please refer to the sample code for detailed usage.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.4.5.6 SUSI_IMC_VPM_SetIgnitionMode

Syntax:

Linux / CE	USHORT **SUSI_IMC_VPM_SetIgnitionMode**(IN IGNITION_MODE nMode);
Android	short **SUSI_IMC_VPM_SetIgnitionMode**(int mode)

Description:

Set the ignition mode of VPM.

Parameters:

nMode [in]

Linux / Windows

Ignition mode.

Android:

Ignition mode. The valid values are enumerated in class IGNITION_MODE .

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

Remark:

None

5.4.5.7 SUSI_IMC_VPM_GetIgnitionMode

Syntax:

Linux / CE	USHORT **SUSI_IMC_VPM_GetIgnitionMode**(OUT PIGNITION_MODE pnMode);
Android	short **SUSI_IMC_VPM_GetIgnitionMode**(int[] pnMode)

Description:

Get the ignition mode of VPM.

Parameters:

nMode [out]

Linux / Windows

Point to read buffer that specifies the ignition mode.

Android:

An allocated array of size 1 for storing the returned value of ignition mode.

The returned value will be stored at index 0.

The returned value should be the values enumerated in class IGNITION_MODE .

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

Remark:

None

5.4.5.8 SUSI_IMC_VPM_SetIgnOnDelay

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_SetIgnOnDelay**(IN USHORT usValue);
Android	short **SUSI_IMC_VPM_SetIgnOnDelay**(short usValue)

Description:

Set the on event delay time of VPM.

Parameters:

usValue [in]

Linux / Windows

Set the ignition ON delay time. The unit is second.

Android:

Set the ignition ON delay time. The unit is second.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows / Android

TREK-674 Linux / Windows

Remark:

The argument unit is the second.

The range of on event delay is from 1 to 18000.

5.4.5.9 SUSI_IMC_VPM_GetIgnOnDelay

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_GetOnDelay** (OUT PUSHORT usValue);
Android	short **SUSI_IMC_VPM_GetIgnOnDelay**(short[] sec)

Description:

Get the on event delay time of VPM.

Parameters:

usValue [out]

Linux / Windows

Get the ignition ON delay time. The unit is second.

Android:

An allocated array of size 1 for storing the returned value of ignition ON delay time in unit of second.

The returned value will be stored at index 0.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

The argument unit is the second.

5.4.5.10 SUSI_IMC_VPM_SetIgnOffEventDelay

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_SetIgnOffEventDelay**(IN USHORT usValue);
Android	short **SUSI_IMC_VPM_SetIgnOffEventDelay**(short usValue)

Description:

Set the off event delay time of VPM.

Parameters:

usValue [in]

Linux / Windows

Set the ignition off delay time. The unit is second.

Android:

Set the ignition off delay time. The unit is second.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

The argument unit is the second.

The range of off event delay is from 1 to 18000.

5.4.5.11 SUSI_IMC_VPM_GetIgnOffEventDelay

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_GetIgnOffEventDelay**(OUT PUSHORT usValue);
Android	short **SUSI_IMC_VPM_GetIgnOffEventDelay**(short[] usValue)

Description:

Get the off event delay time of VPM.

Parameters:

usValue [out]

Linux / Windows

Get the ignition off delay time. The unit is second.

Android:

An allocated array of size 1 for storing the returned value of ignition off delay time.

The returned value will be stored at index 0.

The unit is second.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

The argument unit is the second.

5.4.5.12 SUSI_IMC_VPM_SetIgnHardOffDelay

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_SetIgnHardOffDelay**(IN USHORT usValue);
Android	short **SUSI_IMC_VPM_SetIgnHardOffDelay**(short usValue)

Description:

Set the hard off event delay time of VPM.

Parameters:

usValue [in]

Linux / Windows

Set the hard off delay time. The unit is second.

Android:

Set the hard off delay time. The unit is second.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

The argument unit is the second.

The range of hard off event delay is from 1 to 18000.

5.4.5.13 SUSI_IMC_VPM_GetIgnHardOffDelay

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_GetIgnHardOffDelay**(OUT PUSHORT usValue);
Android	short **SUSI_IMC_VPM_GetIgnHardOffDelay**(short[] sec)

Description:

Get the hard off event delay time of VPM.

Parameters:

usValue [out]

Linux / Windows

Get the hard off delay time. The unit is second.

Android:

An allocated array of size 1 for storing the returned value of hard off delay time.

The returned value will be stored at index 0.

The unit is second.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

The argument unit is the second.

5.4.5.14 SUSI_IMC_VPM_SetIgnSuspendHardOffDelay

Syntax:

Linux / CE	USHORT **SUSI_IMC_VPM_SetIgnSuspendHardOffDelay**(IN USHORT usValue);
Android	short **SUSI_IMC_VPM_SetIgnSuspendHardOffDelay**(short sec)

Description:

Set the suspend mode hard off event delay time of VPM.

Parameters:

usValue [in]

Linux / Windows

Set the suspend mode hard off delay time. The unit is second.

Android:

Set the suspend mode hard off delay time. The unit is second.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

Remark:

The argument unit is the second.

The range of hard off event delay is from 1 to 18000.

5.4.5.15 SUSI_IMC_VPM_GetIgnSuspendHardOffDelay

Syntax:

Linux / CE	USHORT **SUSI_IMC_VPM_GetIgnSuspendHardOffDelay**(OUT PUSHORT usValue);
Android	short **SUSI_IMC_VPM_GetIgnSuspendHardOffDelay**(short[] sec)

Description:

Get the suspend mode hard off event delay time of VPM.

Parameters:

usValue [out]

Linux / Windows

Get the suspend mode hard off time. The unit is second.

Android:

An allocated array of size 1 to store the returned value of suspend mode hard off time in unit of second.

The returned value will be stored at index 0.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

Remark:

The argument unit is the second.

5.4.5.16 SUSI_IMC_VPM_SetForceShutDown

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_SetForceShutDown**(IN BOOL bEnable);

Description:

Enable/Disable force shutdown function. If the function is enabling, the VPM will force shutdown after the delay time escaped. In other word, device wills trigger power off sequence when the force shutdown delay time escaped.

Parameters:

bEnable [in]

Control the force shutdown function.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows, firmware version >= 1.020

TREK-674 Linux / Windows , firmware version >= 1.020

Remark:

True if force shutdown function is used.

5.4.5.17 SUSI_IMC_VPM_GetForceShutDown

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_GetForceShutDown**(OUT BOOL& bEnable);

Description:

Get the current force shutdown function status.

Parameters:

bEnable [out]

Get current force shutdown function status.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows, firmware version >= 1.020

TREK-674 Linux / Windows , firmware version >= 1.020

Remark:

True if force shutdown function is used.

5.4.5.18 SUSI_IMC_VPM_SetForceShutDownDelay

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_SetForceShutDownDelay**(IN USHORT usValue);

Description:

Set the force shutdown event delay time of VPM. When the force shutdown function enable, VPM will countdown the delay time. When the time up, VPM will force run the shutdown flow.

Parameters:

usValue [in]

Set the force shutdown event delay time. The unit is second.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows, firmware version >= 1.020

TREK-674 Linux / Windows , firmware version >= 1.020

Remark:

The argument unit is the second.

The range of on event delay is from 1 to 3600.

5.4.5.19 SUSI_IMC_VPM_GetForceShutDownDelay

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_GetForceShutDownDelay** (OUT PUSHORT usValue);

Description:

Get the force shutdown event delay time of VPM.

Parameters:

usValue [out]

Get the force shutdown event delay time. The unit is second.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows, firmware version >= 1.020

TREK-674 Linux / Windows , firmware version >= 1.020

Remark:

The argument unit is the second.

5.4.5.20 SUSI_IMC_VPM_SetPrePowerOffAlarmDelay

Syntax:

Linux / Windows	USHORT **SUSI_IMC_VPM_SetPrePowerOffAlarmDelay**(IN USHORT usValue);

Description:

Set the prePower-Off alarm event delay time of VPM. User can register a Event to library. When received the Event and prePower-Off Alarm function enable, VPM will countdown the delay time. When the time up, VPM enter normal shutdown flow. In other word, user have an additional period time prepare shutdown process before OS shutdown.

Parameters:

usValue [in]

Set the prePower-Off alarm event delay time. The unit is second.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows, firmware version >= 1.023

TREK-674 Linux / Windows , firmware version >= 1.021

Remark:

The range of on event delay is from 1 to 240.

5.4.5.21 SUSI_IMC_VPM_GetPrePowerOffAlarmDelay

Syntax:

Linux / Windows	USHORT **SUSI_IMC_VPM_GetPrePowerOffAlarmDelay** (OUT PUSHORT usValue);

Description:

Get the prePower-Off alarm event delay time of VPM

Parameters:

usValue [out]

Get theprePower-Off alarm event delay time. The unit is second.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows, firmware version >= 1.023

TREK-674 Linux / Windows , firmware version >= 1.021

5.4.5.22 SUSI_IMC_VPM_SetPrePowerOffAlarm

Syntax:

Linux / Windows	USHORT **SUSI_IMC_VPM_SetPrePowerOffAlarm**(IN BOOL bEnable);

Description:

Enable/Disable prePower-Off notify function. If the function is enabling, the VPM will send a Event to Host and countdown a delay time for user using this time to prepare shutdown process. In other word, device wills trigger power off alarm event when ignition on to off or the car battery voltage too low.

Parameters:

bEnable [in]

Linux / Windows

Control prePower-Off alarm function.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows, firmware version >= 1.023

TREK-674 Linux / Windows , firmware version >= 1.021

5.4.5.23 SUSI_IMC_VPM_GetPrePowerOffAlarm

Syntax:

Linux / Windows	USHORT **SUSI_IMC_VPM_GetPrePowerOffAlarm**(OUT BOOL& bEnable);

Description:

Get the current prePower-Off alarm function status.

Parameters:

bEnable [out]

Linux / Windows

Get current prePower-Off alarm status. 1 is Enable, 0 is Disable.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows, firmware version >= 1.023

TREK-674 Linux / Windows , firmware version >= 1.021

5.4.5.24 SUSI_IMC_VPM_SetIgnPowerOffEvent

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_SetIgnPowerOffEvent** (IN void* Event);

Description:

Set an Event handler to library to listen Ignition On to Off prePower-Off alarm event. Event object is a HANDLE on Windows and the event object is pthread_cond_t on Linux.

Parameters:

hEvent [in]

Event handler of prePower-Off alarm event.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows, firmware version >= 1.023

TREK-674 Linux / Windows , firmware version >= 1.021

5.4.5.25 SUSI_IMC_VPM_SetLBPPowerOffEvent

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_SetLBPPowerOffEvent** (IN void* Event);

Description:

Set an Event handler to library to listen Car Low Battery prePower-Off alarm event. Event object is a HANDLE on Windows and the event object is pthread_cond_t on Linux.

Parameters:

hEvent [in]

Event handler of prePower-Off alarm event.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows, firmware version >= 1.023

TREK-674 Linux / Windows , firmware version >= 1.021

5.4.5.26 SUSI_IMC_VPM_SetLBPLowDelay

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_SetLBPLowDelay**(IN USHORT usValue);
Android	short **SUSI_IMC_VPM_SetLBPLowDelay**(short sec)

Description:

Set the low event delay time of VPM.

Parameters:

usValue [in]

Linux / Windows

Set the low event delay time. The unit is second.

Android:

Set the low event delay time. The unit is second.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

The argument unit is the second.

The range of low event delay is from 1 to 3600.

5.4.5.27 SUSI_IMC_VPM_GetLBPLowDelay

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_GetLBPLowDelay**(OUT PUSHORT usValue);
Android	short **SUSI_IMC_VPM_GetLBPLowDelay**(short[] sec)

Description:

Get the low event delay time of VPM.

Parameters:

usValue [out]

Linux / Windows

Get the low event delay time. The unit is second.

Android:

An allocated array of size 1 for storing the returned value of low event delay time.

The returned value will be stored at index 0.

The unit is second.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

The argument unit is the second.

5.4.5.28 SUSI_IMC_VPM_SetLBPLowHardDelay

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_SetLBPLowHardDelay**(IN USHORT usValue);
Android	short **SUSI_IMC_VPM_SetLBPLowHardDelay**(short sec)

Description:

Set the low hard event delay time of VPM.

Parameters:

usValue [in]

Linux / Windows

Set the low hard event delay time. The unit is second.

Android:

Set the low hard event delay time. The unit is second.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

The argument unit is the second.

The range of low hard event delay is from 1 to 3600.

5.4.5.29 SUSI_IMC_VPM_GetLBPLowHardDelay

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_GetLBPLowHardDelay**(OUT PUSHORT usValue);
Android	short **SUSI_IMC_VPM_GetLBPLowHardDelay**(short[] sec)

Description:

Get the low hard event delay time of Power Management.

Parameters:

usValue [out]

Linux / Windows

Get the low hard event delay time. The unit is second.

Android:

An allocated array of size 1 for storing the returned value of low hard event delay time.

The return value will be stored at index 0.

The unit is second.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

The argument unit is the second.

5.4.5.30 SUSI_IMC_VPM_SetLBPThresholdFor12VSystem

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_SetLBPThresholdFor12VSystem**(IN FLOAT fVoltage);
Android	short **SUSI_IMC_VPM_SetLBPThresholdFor12VSystem**(float voltage)

Description:

Set the threshold of 12 voltage battery.

Parameters:

fVoltage [in]

Linux / Windows

Set the post-boot LBP threshold. The unit is Volt.

Android:

Set the post-boot LBP threshold. The unit is Volt.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

The range of 12 battery voltage is from 10.11V to 12.26V.

5.4.5.31 SUSI_IMC_VPM_GetLBPThresholdFor12VSystem

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_GetLBPThresholdFor12VSystem**(OUT PFLOAT fVoltage);
Android	short **SUSI_IMC_VPM_GetLBPThresholdFor12VSystem**(float[] voltage)

Description:

Get the threshold of 12 voltage battery.

Parameters:

fVoltage [out]

Linux / Windows

Get the post-boot LBP threshold. The unit is Volt.

Android:

An allocated array of size 1 for storing the returned value of post-boot LBP threshold.

The return value will be stored at index 0.

The unit is Volt.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

The range of 12 battery voltage is from 10.11V to 12.26V.

5.4.5.32 SUSI_IMC_VPM_SetLBPThresholdFor24VSystem

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_SetLBPThresholdFor24VSystem**(IN FLOAT fVoltage);
Android	short **SUSI_IMC_VPM_SetLBPThresholdFor24VSystem**(float voltage)

Description:

Set the threshold of 24 voltage battery.

Parameters:

fVoltage [in]

Linux / Windows

Set the post-boot LBP threshold. The unit is Volt.

Android:

Set the post-boot LBP threshold. The unit is Volt.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

The range of 24 battery voltage is from 21.09V to 23.29V.

5.4.5.33 SUSI_IMC_VPM_GetLBPThresholdFor24VSystem

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_GetLBPThresholdFor24VSystem**(OUT PFLOAT fVoltage);
Android	short **SUSI_IMC_VPM_GetLBPThresholdFor24VSystem**(float[] voltage)

Description:

Get the threshold of 24 voltage battery.

Parameters:

fVoltage [out]

Linux / Windows

Get the post-boot LBP threshold. The unit is Volt.

Android:

An allocated array of size 1 for storing the returned value of post-boot LBP threshold.

The return value will be stored at index 0.

The unit is Volt.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

The range of 24 battery voltage is from 21.09V to 23.29V.

5.4.5.34 SUSI_IMC_VPM_SetPrebootThresholdFor12VSystem

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_SetPrebootThresholdFor12VSystem**(IN FLOAT fVoltage);
Android	short **SUSI_IMC_VPM_SetPrebootThresholdFor12VSystem**(float voltage)

Description:

Set the preboot threshold of 12 voltage battery.

Parameters:

fVoltage [in]

Linux / Windows

Set the preboot LBP threshold. The unit is Volt.

Android:

Set the preboot LBP threshold. The unit is Volt.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

The range of 12 battery voltage is from 10.11V to 12.26V.

5.4.5.35 SUSI_IMC_VPM_GetPrebootThresholdFor12VSystem

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_GetPrebootThresholdFor12VSystem**(OUT PFLOAT pfVoltage);
Android	short **SUSI_IMC_VPM_GetPrebootThresholdFor12VSystem**(float[] voltage)

Description:

Get the preboot threshold of 12 voltage battery.

Parameters:

fVoltage [out]

Linux / Windows

Get the pret-boot LBP threshold. The unit is Volt.

Android:

An allocated array of size 1 for storing the returned value of pre-boot LBP threshold.

The return value will be stored at index 0.

The unit is Volt.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

The range of 24 battery voltage is from 21.09V to 23.29V.

5.4.5.36 SUSI_IMC_VPM_SetPrebootThresholdFor24VSystem

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_SetPrebootThresholdFor24VSystem**(IN FLOAT fVoltage);
Android	short **SUSI_IMC_VPM_SetPrebootThresholdFor24VSystem**(float voltage)

Description:

Set the preboot threshold of 24 voltage battery.

Parameters:

fVoltage [in]

Linux / Windows

Set the preboot threshold. The unit is Volt.

Android:

Set the preboot threshold. The unit is Volt.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

The range of 24 battery voltage is from 21.09V to 23.29V.

5.4.5.37 SUSI_IMC_VPM_GetPrebootThresholdFor24VSystem

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_GetPrebootThresholdFor24VSystem**(OUT PFLOAT pfVoltage);
Android	short **SUSI_IMC_VPM_GetPrebootThresholdFor24VSystem**(float[] voltage)

Description:

Get the preboot threshold of 24 voltage battery.

Parameters:

fVoltage [out]

Linux / Windows

Get the preboot LBP threshold. The unit is Volt

Android:

An allocated array of size 1 for storing the returned value of pre-boot LBP threshold.

The return value will be stored at index 0.

The unit is Volt. .

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

The range of 24 battery voltage is from 21.09V to 23.29V.

5.4.5.38 SUSI_IMC_VPM_GetLBPThresholdInfo

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_GetLBPThresholdInfo**(IN CAR_BATTERY_MODE nCarBatteryMode, OUT PBATTERY_INFO BatteryInfo);
Android	short **SUSI_IMC_VPM_GetLBPThresholdInfo**(int carBatteryMode, BATTERY_INFO batteryInfo)

Description:

Get the threshold of programmable voltage battery.

Parameters:

nCarBatteryMode [in]

Linux / Windows

The battery mode.

Android

The battery mode. Available value is enumerated in class CAR_BATTERY_MODE

BatteryInfo [out]

Linux / Windows

Get the battery info.

Android

A instance of class BATTERY_INFO for storing returned values of LBP threshold information.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

The range of programmable battery voltage.

5.4.5.39 SUSI_IMC_VPM_GetIgnitionStatus

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_GetIgnitionStatus**(OUT PBOOL bIngON);
Android	short **SUSI_IMC_VPM_GetIgnitionStatus**(boolean[] bIngON)

Description:

Get the ignition status.

Parameters:

bIgnON [out]

Linux / Windows

Get the ignition status.

Android

An allocated array of size 1 for storing the returned value of ignition status.

The return value will be stored at index 0.

TRUE, if ignition status is ON.

FALSE, if ignition status is OFF.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

True if the ignition status is ON.

5.4.5.40 SUSI_IMC_VPM_GetCarBatteryVoltage

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_GetCarBatteryVoltage**(OUT PFLOAT fVoltage);
Android	short **SUSI_IMC_VPM_GetCarBatteryVoltage**(float[] voltage)

Description:

Get current battery voltage.

Parameters:

fVoltage [out]

Linux / Windows

Get the battery voltage.

Android

An allocated array of size 1 for storing the returned value of battery voltage.

The return value will be stored at index 0.

The unit is Volt. .

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

The argument unit is voltage.

5.4.5.41 SUSI_IMC_VPM_SetCurrentCarBatteryMode

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_SetCurrentCarBatteryMode** (CAR_BATTERY_MODE nCarBatterMode);
Android	short **SUSI_IMC_VPM_SetCurrentCarBatteryMode**(int carBatteryMode)

Description:

Set the current battery mode, the battery mode affects the low battery protection's threshold. Each mode have the difference threshold value and range.

Parameters:

nCarBatterMode [in]

Linux / Windows

Set the battery mode.

Android:

Set the battery mode. Available value is enumerated in class CAR_BATTERY_MODE

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows / Android, firmware version >= 1.021

TREK-674 Linux / Windows, firmware version >= 1.020

Remark:

When you call the **SUSI_IMC_VPM_SetCurrentCarBatteryMode** the low battery protect function will force disable to avoid the target machine can't boot issue. If you want using the low battery protect function, you need manual enable the function again after change the battery mode.

5.4.5.42 SUSI_IMC_VPM_GetCurrentCarBatteryMode

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_GetCurrentCarBatteryMode** (OUT CAR_BATTERY_MODE& nCarBatterMode);
Android	short **SUSI_IMC_VPM_GetCurrentCarBatteryMode**(int[] carBatteryMode)

Description:

Get the current battery mode, the battery mode affects the low battery protection's threshold. Each mode have the difference threshold value and range.

Parameters:

nCarBatterMode [out]

Linux / Windows

Get the battery mode.

Android

An allocated array of size 1 for storing the returned value of battery mode.

The return value will be stored at index 0.

Available value is enumerated in class CAR_BATTERY_MODE

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.4.5.43 SUSI_IMC_VPM_SetLBPStatus

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_SetLBPStatus**(IN BOOL bEnable);
Androi	short **SUSI_IMC_VPM_SetLBPStatus**(boolean bEnable)

Description:

Enable/Disable low battery protection function. If the function is enabling, the VPM will open the postboot protection flow. In other word, device wills trigger power off sequence when the power voltage low than the defined threshold.

Parameters:

bEnable [in]

Linux / Windows

Control the low battery protection function.

Android:

Control the low battery protection function.

TRUE for enable.

FALSE for disable.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

True if low battery protection is used.

5.4.5.44 SUSI_IMC_VPM_GetLBPStatus

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_GetLBPStatus**(OUT BOOL& bEnable);
Android	short **SUSI_IMC_VPM_GetLBPStatus**(boolean[] bEnable)

Description:

Get the current low battery protection status.

Parameters:

bEnable [out]

Linux / Windows

Get current low battery protection status.

Android

An allocated array of size 1 for storing the returned value of current low battery protection status.

The return value will be stored at index 0.

TRUE for enable.

FALSE for disable.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

True if low battery protection is used.

5.4.5.45 SUSI_IMC_VPM_SetPrebootLBPStatus

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_SetPrebootLBPStatus**(IN BOOL bEnable);
Android	short **SUSI_IMC_VPM_SetPrebootLBPStatus**(boolean status)

Description:

Enable/Disable preboot low battery protection function. If the function is enabling, the VPM will open the preboot protection flow. In other word, device check the voltage when boot up time. If the power voltage low than the defined threshold, the device won't be open.

Parameters:

bEnable [in]

Linux / Windows

Control preboot low battery protection function

Android:

Control the preboot low battery protection function.

TRUE for enable.

FALSE for disable..

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

True if low battery protection is used.

5.4.5.46 SUSI_IMC_VPM_GetPrebootLBPStatus

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_GetPrebootLBPStatus**(OUT PBOOL pbEnable);
Android	short **SUSI_IMC_VPM_GetPrebootLBPStatus**(boolean[] status)

Description:

Get the preboot system low battery protection status.

Parameters:

bEnable [out]

Linux / Windows

Get the preboot system low battery protection status.

Android

An allocated array of size 1 for storing the returned value of current perboot low battery protection status.

The return value will be stored at index 0.

TRUE for enable.

FALSE for disable.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

True if low battery protection is used.

5.4.5.47 SUSI_IMC_VPM_SetATMode

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_SetATMode**(IN BOOL bEnable);
Android	short **SUSI_IMC_VPM_SetATMode**(boolean status)

Description:

Enable/Disable the AT mode. The device will be opened at the ignition on and the first time power supply.

Parameters:

bEnable [in]

Linux / Windows

Control the AT mode.

Android:

Control the AT mode

TRUE for enable.

FALSE for disable.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.4.5.48 SUSI_IMC_VPM_GetATMode

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_GetATMode**(OUT BOOL& bEnable);
Android	short **SUSI_IMC_VPM_GetATMode**(boolean[] status)

Description:

Get the AT mode.

Parameters:

bEnable [out]

Linux / Windows

Get the AT mode.

Android

An allocated array of size 1 for storing the returned value of current AT mode status.

The return value will be stored at index 0.

TRUE for enable.

FALSE for disable.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.4.5.49 SUSI_IMC_VPM_SetKeepAliveMode

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_SetKeepAliveMode**(IN BOOL bEnable);
Android	short **SUSI_IMC_VPM_SetKeepAliveMode**(boolean status)

Description:

Enable/Disable Keep Alive mode. The device will maintenance the power on state when the ignition on.

Parameters:

bEnable [in]

Linux / Windows

Control Keep Alive mode.

Android:

Control Keep Alive mode.

TRUE for enable.

FALSE for disable.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.4.5.50 SUSI_IMC_VPM_GetKeepAliveMode

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_GetKeepAliveMode**(OUT BOOL& bEnable);
Android	short **SUSI_IMC_VPM_GetKeepAliveMode**(boolean[] status)

Description:

Get the keep alive mode status.

Parameters:

bEnable [in]

Linux / Windows

Get the Keep Alive mode status

Android

An allocated array of size 1 for storing the returned value of current Keep Alive mode status.

The return value will be stored at index 0.

TRUE for enable.

FALSE for disable.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.4.5.51 SUSI_IMC_VPM_SetWakeupSourceControlStatus

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_SetWakeupSourceControlStatus**(IN WAKEUP_SOURCE wakeup_source, IN BOOL bEnable);
Android	short **SUSI_IMC_VPM_SetWakeupSourceControlStatus**(int wakeupSource, boolean bEnable)

Description:

Enable/Disable wakeup source.

Parameters:

wakeup_source [in]

Linux / Windows

Set wakeup source.

Android:

Set wakeup source. Available value is enumerated in class WAKEUP_SOURCE

bEnable [in]

Linux / Windows

Enable/Disable wakeup source.

Android:

Enable/Disable wakeup source.

TRUE for enable.

FALSE for disable.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

Different Platform support different wakeup source. Example, TREK-520 didn't to support alarm, hotkey, d2 and d3. The detail can to reference the product manual.

5.4.5.52 SUSI_IMC_VPM_GetWakeupSourceControlStatus

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_GetWakeupSourceControlStatus**(IN WAKEUP_SOURCE wakeup_source, OUT PBOOL pbEnable);
Android	short **SUSI_IMC_VPM_GetWakeupSourceControlStatus**(int wakeupSource, boolean[] status)

Description:

Get the status of wakeup source.

Parameters:

wakeup_source [in]

Linux / Windows

The wakeup source.

Android:

The wakeup source. Available value is enumerated in class WAKEUP_SOURCE

pbEnable [out]

Linux / Windows

Get the status of wakeup source

Android

An allocated array of size 1 for storing the returned value of current wakeup source status.

The return value will be stored at index 0.

TRUE for enable.

FALSE for disable.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.4.5.53 SUSI_IMC_VPM_SetShutdownSourceControlStatus

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_SetShutdownSourceControlStatus**(IN SHUTDOWN_SOURCE shutdown_source, IN BOOL bEnable);
Android	short **SUSI_IMC_VPM_SetShutdownSourceControlStatus**(int shutdownSource, boolean bEnable)

Description:

Enable/Disable shutdown source. Default behavior the computer can shutdown by ignition off, Power Button or Software shutdown command. User can use this function to disable the shutdown source. Shutdown source can't disable all for prevent the computer stuck when the software crash.

Parameters:

shutdown_source [in]

Linux / Windows

Set shutdown source.

Android:

Set shutdown source. Available value is enumerated in class SHUTDOWN_SOURCE

bEnable [in]

Linux / Windows

Enable/Disable shutdown source.

Android:

Enable/Disable shutdown source.

TRUE for enable.

FALSE for disable.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows / Android, firmware version >= 1.020

TREK-572 Linux / Windows, firmware version >= 1.020

TREK-674 Linux / Windows, firmware version >= 1.020

5.4.5.54 SUSI_IMC_VPM_GetShutdownSourceControlStatus

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_GetShutdownSourceControlStatus**(IN SHUTDOWN_SOURCE shutdown_source, OUT PBOOL pbEnable);
Android	short **SUSI_IMC_VPM_GetShutdownSourceControlStatus**(int shutdownSource, boolean[] status)

Description:

Get the status of shutdown source.

Parameters:

shutdown_source [in]

Linux / Windows

The shutdown source.

Android:

The shutdown source. Available value is enumerated in class SHUTDOWN_SOURCE

pbEnable [out]

Linux / Windows

Get the status of shutdown source

Android

An allocated array of size 1 for storing the returned value of current wakeup source status.

The return value will be stored at index 0.

TRUE for enable.

FALSE for disable.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows / Android, firmware version >= 1.020

TREK-572 Linux / Windows, firmware version >= 1.020

TREK-674 Linux / Windows, firmware version >= 1.020

Remark:

None

5.4.5.55 SUSI_IMC_VPM_GetLastWakeupSource

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_GetLastWakeupSource**(OUT PWAKEUP_SOURCE pWakeup_source);
Android	short **SUSI_IMC_VPM_GetLastWakeupSource**(int[] wakeupSource)

Description:

Get the last wakeup source from VPM.

Parameters:

pWakeup_source [out]

Linux / Windows

Get the last wakeup source.

Android:

Get the last wakeup source. Available value is enumerated in class WAKEUP_SOURCE

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.4.5.56 SUSI_IMC_VPM_SetRTCTime

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_SetRTCTime** (IN VPM_RTC_TIME nRTC_time);
Android	short **SUSI_IMC_VPM_SetRTCTime**(VPM_RTC_TIME rtcTime)

Description:

Set the external RTC time. The RTC time will affects the alarm wakeup time.

Parameters:

PVPM_RTC_TIME [in]

Linux / Windows

Set the RTC time info.

Android

A instance of class VPM_RTC_TIME which containing the desired RTC time.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.4.5.57 SUSI_IMC_VPM_GetRTCTime

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_GetRTCTime** (OUT PVPM_RTC_TIME pnRTC_time);
Android	short **SUSI_IMC_VPM_GetRTCTime**(VPM_RTC_TIME rtcTime)

Description:

Get the RTC time.

Parameters:

PVPM_RTC_TIME [out]

Linux / Windows

Get the RTC time information.

Android

A instance of class VPM_RTC_TIME for storing the returned time.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.4.5.58 SUSI_IMC_VPM_SetAlarmActive

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_SetAlarmActive**(IN BOOL bActive);
Android	short **SUSI_IMC_VPM_SetAlarmActive**(boolean bActive)

Description:

Enable/Disable alarm wakeup function. The same function with SetWakeupSourceControlStatus(WAKEUP_SOURCE_ALARM, bEnable)

Parameters:

bActive [in]

Linux / Windows

Enable/Disable Alarm status.

Android:

Enable/Disable Alarm status.

TRUE for enable.

FALSE for disable.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

To make alarm as wakeup source, you should set the wakeup source of alarm to ENABLE by using API SUSI_IMC_VPM_SetWakeupSourceControlStatus()

5.4.5.59 SUSI_IMC_VPM_GetAlarmActive

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_GetAlarmActive**(OUT BOOL& bActive);
Android	short **SUSI_IMC_VPM_GetAlarmActive**(boolean[] bActive)

Description:

Get alarm wakeup status.

Parameters:

bActive [out]

Linux / Windows

Alarm status.

Android

An allocated array of size 1 for storing the returned value of alarm status. .

The return value will be stored at index 0.

TRUE for enable.

FALSE for disable.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.4.5.60 SUSI_IMC_VPM_SetAlarmTime

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_SetAlarmTime**(IN ALARM_MODE Alarm_mode, IN VPM_ALARM_TIME Alarm_time);
Android	short **SUSI_IMC_VPM_SetAlarmTime**(int alarmMode, VPM_ALARM_TIME alarmTime)

Description:

Set the alarm wakeup time.

For example 1, if you want boot at 13:45 each day. Set the ALARM_MODE= DAILY, nHour=13, nMinute=45, nDayOfWeek=don't care

For example 2, if you want boot at 4:05 each Tuesday. Set the ALARM_MODE= WEEKLY, nHour=4, nMinute=5, nDayOfWeek=2

Parameters:

Alarm_mode [in]

Linux / Windows

Alarm wakeup mode.

Android:

Alarm wakeup mode. Available value is enumerated in class ALARM_MODE .

Alarm_time [in]

Linux / Windows

The structure of alarm time.

Android

A instance of class VPM_ALARM_TIME which containing the desired alarm time.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

NO_ALARM only using SUSI_IMC_VPM_GetAlarmTime

5.4.5.61 SUSI_IMC_VPM_GetAlarmTime

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_GetAlarmTime**(OUT PALARM_MODE pnAlarm_mode, OUT PVPM_ALARM_TIME pnAlarm_time);
Android	short **SUSI_IMC_VPM_GetAlarmTime**(int[] alarmMode, VPM_ALARM_TIME alarmTime)

Description:

Get the alarm time.

Parameters:

Alarm_mode [out]

Linux / Windows

Alarm wakeup mode.

Android:

An allocated array of size 1 for storing the returned value of current alarm wakeup mode.

The return value will be stored at index 0.

Available value is enumerated in class ALARM_MODE **.**

Alarm_time [out]

Linux / Windows

The structure of alarm time.

Android

A instance of class VPM_ALARM_TIME for storing the returned values of alarm time.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.4.5.62 SUSI_IMC_VPM_GetBackupBatteryVoltage

Syntax:

Linux / CE	USHORT **SUSI_IMC_VPM_GetBackupBatteryVoltage**(OUT PFLOAT pfVoltage);
Android	short **SUSI_IMC_VPM_GetBackupBatteryVoltage**(float[] voltage)

Description:

Get current backup battery voltage.

Parameters:

pfVoltage [out]

Linux / Windows

Get battery voltage.

Android

An allocated array of size 1 for storing the returned value of battery voltage

The return value will be stored at index 0.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

Remark:

The argument unit is voltage.

5.4.5.63 SUSI_IMC_VPM_GetBackupBatteryRemainingCapacity

Syntax:

Linux / CE	USHORT **SUSI_IMC_VPM_GetBackupBatteryRemainingCapacity**(OUT PINT piValue);
Android	short **SUSI_IMC_VPM_GetBackupBatteryRemainingCapacity**(int[] capacity)

Description:

Get the current backup battery compensated capacity remaining.

Parameters:

piValue [out]

Linux / Windows

Get current backup battery remaining capacity.

Android

An allocated array of size 1 for storing the returned value of current backup battery remaining capacity.

The return value will be stored at index 0.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

Remark:

The argument unit is mAh.

5.4.5.64 SUSI_IMC_VPM_GetBackupBatteryPercentageOfCharge

Syntax:

Linux / CE	USHORT **SUSI_IMC_VPM_GetBackupBatteryPercentageOfCharge**(OUT PINT piValue);
Android	short **SUSI_IMC_VPM_GetBackupBatteryPercentageOfCharge**(int[] piValue)

Description:

Get the current backup battery value of the predicted remaining battery capacity expressed as a percentage of full capacity, with a range of 0 to 100%.

Parameters:

piValue [out]

Linux / Windows

Get backup battery percentage of charge.

Android

An allocated array of size 1 for storing the returned value of backup battery percentage of charge.

The return value will be stored at index 0.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

Remark:

The argument unit is %.

5.4.5.65 SUSI_IMC_VPM_GetBackupBatteryTemperture

Syntax:

Linux / CE	USHORT **SUSI_IMC_VPM_GetBackupBatteryTemperture**(OUT PFLOAT pfTemperture);
Android	short **SUSI_IMC_VPM_GetBackupBatteryTemperture**(float[] temperature)

Description:

Get the current backup battery temperature.

Parameters:

pfTemperture [out]

Linux / Windows

Get the current backup battery temperature.

Android

An allocated array of size 1 for storing the returned value of backup battery temperature.

The return value will be stored at index 0.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

Remark:

The argument unit is %.

5.4.5.66 SUSI_IMC_VPM_GetBackupBatteryRemainingTime

Syntax:

Linux / CE	USHORT **SUSI_IMC_VPM_GetBackupBatteryRemainingTime**(OUT PINT piTime);
Android	short **SUSI_IMC_VPM_GetBackupBatteryRemainingTime**(int[] piTime)

Description:

Get the current selected battery predicted remaining battery life at the present rate of discharge.

Parameters:

piTime [out]

Linux / Windows

Get backup battery remaining time.

Android

An allocated array of size 1 for storing the returned value of backup battery remaining time.

The return value will be stored at index 0.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

Remark:

The argument unit is minutes.

5.4.5.67 SUSI_IMC_VPM_GetBackupBatteryTimeToFull

Syntax:

Linux / CE	USHORT **SUSI_IMC_VPM_GetBackupBatteryTimeToFull**(OUT PINT piTime);
Android	short **SUSI_IMC_VPM_GetBackupBatteryTimeToFull**(int[] piTime)

Description:

Get the current selected battery predicted remaining time until the battery reaches full charge.

Parameters:

piTime [out]

Linux / Windows

Get backup battery remaining time.

Android

An allocated array of size 1 for storing the returned value of backup battery remaining time.

The return value will be stored at index 0.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

Remark:

The argument unit is minutes.

5.4.5.68 SUSI_IMC_VPM_GetBackupBatteryInfo

Syntax:

Linux / CE	USHORT **SUSI_IMC_VPM_GetBackupBatteryInfo**(PBACKUP_BATTERY_INFO pnBackup_battery_info);
Android	short **SUSI_IMC_VPM_GetBackupBatteryInfo**(BACKUP_BATTERY_INFO backupBatteryInfo)

Description:

Get the backup battery information.

Parameters:

PBACKUP_BATTERY_INFO [out]

Linux / Windows

Get the backup battery information.

Android

A instance of class **BATTERY_INFO** for storing the returned values of alarm time.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

Remark:

None

5.4.5.69 SUSI_IMC_VPM_LoadDefaultValue

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_LoadDefaultValue**(void);
Android	short **SUSI_IMC_VPM_LoadDefaultValue**()

Description:

Load the VPM configure to default value.

Parameters:

None

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.4.5.70 SUSI_IMC_VPM_ForceShutdown

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VPM_ForceShutdown**(void);
Android	short **SUSI_IMC_VPM_ForceShutdown**()

Description:

Immediately let VPM enter to power off flow. VPM will immediately send Power-off signal to OS.

Parameters:

None

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows / Android, firmware version >= 1.020

TREK-572 Linux / Windows, firmware version >= 1.020

TREK-674 Linux / Windows, firmware version >= 1.020

Remark:

None

5.5 WATCHDOG

5.5.1 Usage Overview

VPM provides Watchdog functions for you to deal with unexpected system hang. Watchdog is a countdown timer held by VPM.

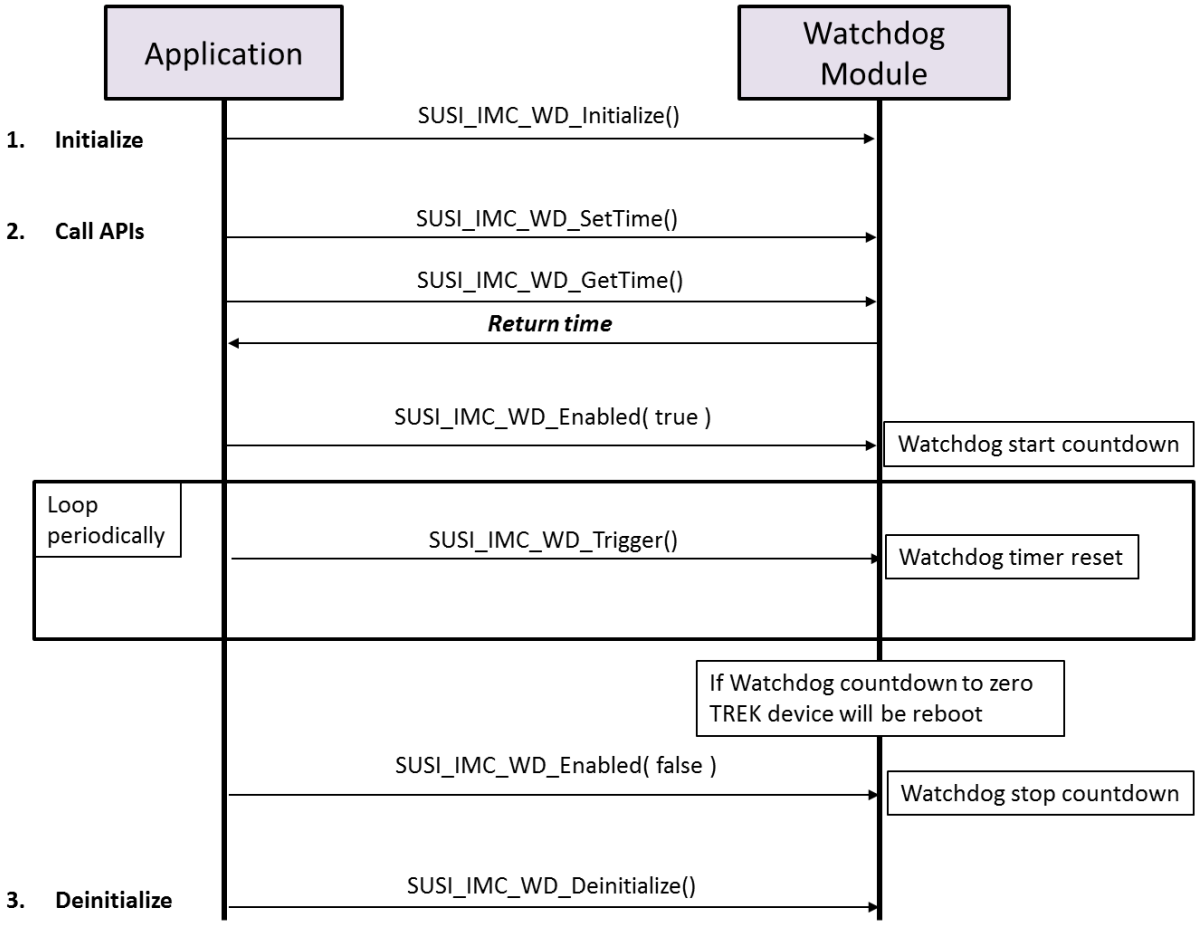
Watchdog API modules provides APIs to control watchdog. You can use the APIs to set desired count down time and to start/stop the watchdog timer. Once the watchdog timer is started, you can call the "trigger" API in your APP periodically to inform watchdog that the system is still working. If the watchdog is "triggered", the timer will be reset and restart counting. When the watchdog timer counts to zero, VPM will force power off and reboot the TREK device to avoid system hang.

The following are the APIs to control the watchdog:

- SUSI_IMC_WD_SetTime() - Set timer timeout value.
- SUSI_IMC_WD_GetTime() - Get timer timeout value.
- SUSI_IMC_WD_Enabled() - Ask watchdog to start/stop countdown.
- SUSI_IMC_WD_Trigger() - Ask watchdog to reset timer and restart countdown.

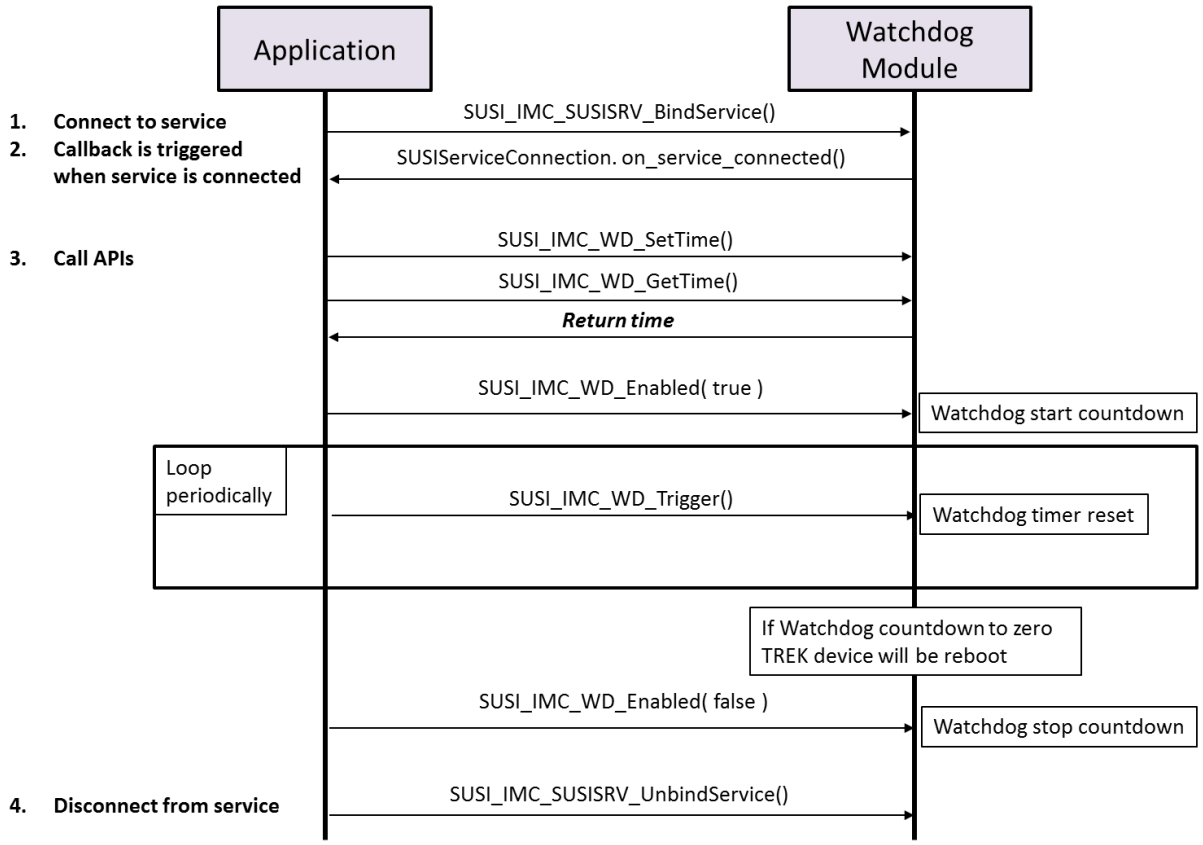
The following parts describe the API usage of Watchdog functions

5.5.1.1 Windows/Linux



For all Watchdog APIs, you should call SUSI_IMC_WD_Initialize() before using the other APIs. To stop the Watchdog module, you must use SUSI_IMC_WD_Deinitialize() to deinitialize.

5.5.1.2 Android



The class for Watchdog APIs for Android is **mrm.client.SUSIServiceClient**.

For the usage, please also refer to the section SUSI SRV Usage Overview.

5.5.2 APIs

5.5.2.1 SUSI_IMC_WD_Initialize

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_WD_Initialize**(IN BYTE port_number);

Description:

Initialize the watchdog library.

Parameters:

port_number [in]

Specifies the port number of watch dog.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

- It shown the UART mapping table as following:

Hardware Platform	OS Platform	Port ID/PATH
TREK-520	WinCE	6
TREK-520	Linux	/proc/vpm_serial
TREK-674 TREK-572	Windows	1
TREK-674 TREK-572	Linux	/dev/ttyS0
TREK-570	Windows	4
TREK-570	Linux	/dev/ttyS3

Note: Linux ignore the port_number argument and using the variable WDT_PATH which defined in /etc/susi/susi_config

5.5.2.2 SUSI_IMC_WD_Deinitialize

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_WD_Deinitialize**(void);

Description:

Deinitialize the watchdog library.

Parameters:

None

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.5.2.3 SUSI_IMC_WD_GetLibVersion

Syntax:

CE / Windows	USHORT **SUSI_IMC_WD_GetLibVersion**(OUT BYTE version[18]);

Description:

Get the library version from TREK SDK.

Parameters:

pVersion [out]

Point to read buffer that specifies the watch dog library's current version.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 CE

TREK-570 Windows

TREK-572 Windows

TREK-674 Windows

Remark:

The version buffer size must be greater than 18 bytes.

5.5.2.4 SUSI_IMC_WD_GetRange

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_WD_GetRange**(DWORD *minimum, DWORD *maximum);
Android	short **SUSI_IMC_WD_GetRange**(int[] min, int[] max)

Description:

Get the minimum and maximum values of the watchdog timer.

Parameters:

minimum [out]

Linux / Windows

Pointer to a variable containing the minimum configurable timeout value in seconds.

Android

An allocated array of size 1 for storing the returned value of minimum configurable timeout value.

The return value will be stored at index 0.

The unit is second.

maximum [out]

Linux / Windows

Pointer to a variable containing the maximum configurable timeout value in seconds.

Android

An allocated array of size 1 for storing the returned value of max configurable timeout value.

The return value will be stored at index 0.

The unit is second.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.5.2.5 SUSI_IMC_WD_SetTime

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_WD_SetTime**(unsigned short Time);
Android	short **SUSI_IMC_WD_SetTime**(int time)

Description:

Set value of watchdog timer.

Parameters:

Time [in]

Linux / Windows

The watchdog timeout value. The unit is second.

Android:

The watchdog timeout value. The unit is second.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

The unit is seconds.

5.5.2.6 SUSI_IMC_WD_GetTime

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_WD_GetTime**(unsigned short* Time);
Android	short SUSI_IMC_WD_GetTime(int[] time)

Description:

Get value of watchdog timer.

Parameters:

Time [out]

Linux / Windows

Pointer to a variable containing the timeout value in seconds.

Android

An allocated array of size 1 for storing the returned value of timeout value.

The return value will be stored at index 0.

The unit is second.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

The unit is seconds.

5.5.2.7 SUSI_IMC_WD_Enabled

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_WD_Enabled**(bool bActive);
Android	short **SUSI_IMC_WD_Enabled**(boolean active)

Description:

Control watchdog timer active status.

Parameters:

bActive [in]

Linux / Windows

Watchdog status

True : Enable watchdog timer.

False : Disable watchdog timer.

Android:

Watchdog status

True : Enable watchdog timer.

False : Disable watchdog timer.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.5.2.8 SUSI_IMC_WD_Trigger

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_WD_Trigger**(void);
Android	short **SUSI_IMC_WD_Trigger**()

Description:

Reload the watchdog timer to the timeout value given in SUSI_IMC_WD_SetTime to prevent the system from rebooting.

Parameters:

None

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

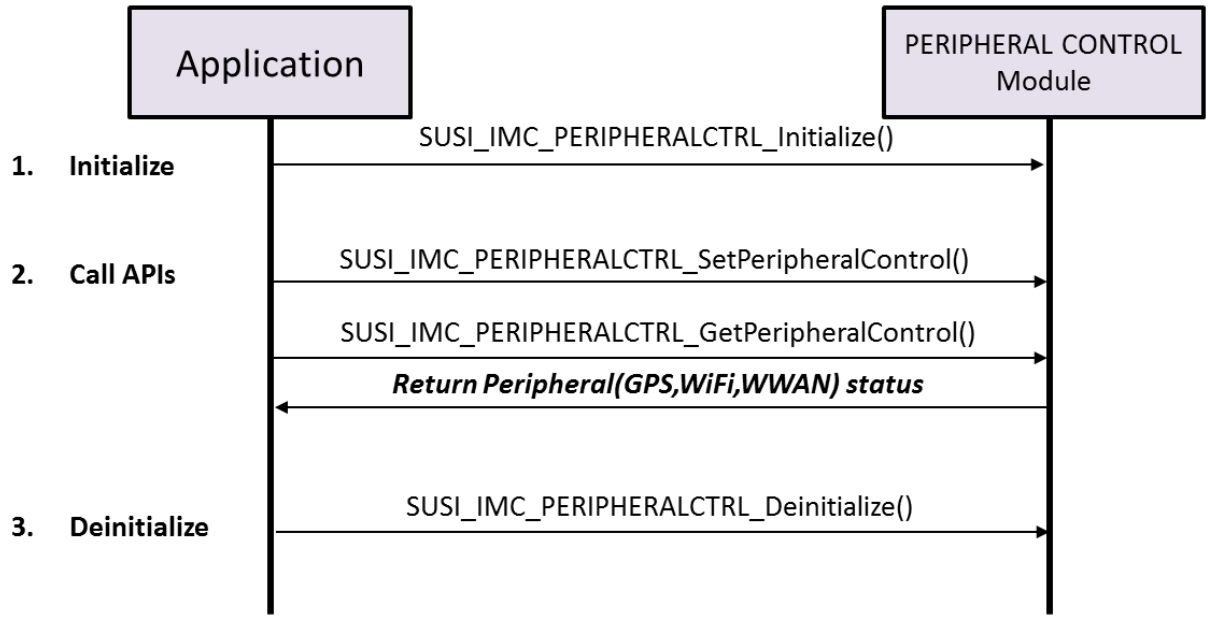
5.6 PERIPHERAL CONTROL

5.6.1 Usage Overview

VPM is able to control the power of peripheral devices (GPS,WiFi,WWAN,etc.). The PERIPHERAL CONTROL module provides APIs to access these functions.

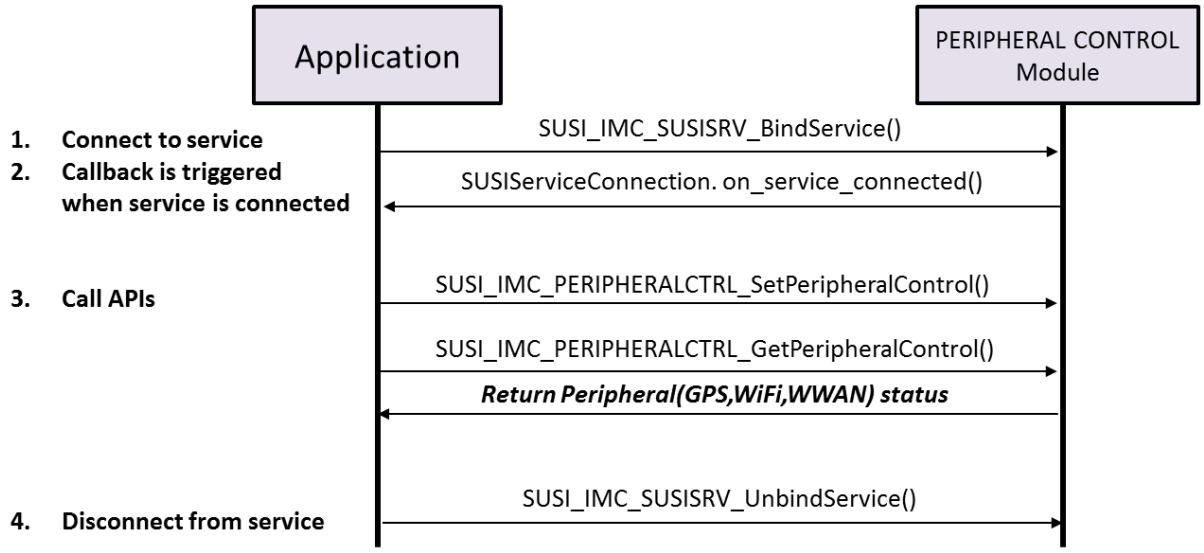
The following parts describe the API usage of PERIPHERAL CONTROL functions.

5.6.1.1 Windows/Linux



For all PERIPHERAL CONTROL APIs, you should call SUSI_IMC_PERIPHERALCTRL_Initialize() before using the other APIs. To stop the module, you must use SUSI_IMC_PERIPHERALCTRL_Deinitialize() to deinitialize.

5.6.1.2 Android



The class for PERIPHERAL CONTROL APIs for Android is **mrm.client.SUSIServiceClient**.

For the usage, please also refer to the section SUSI SRV Usage Overview.

5.6.2 Peripheral Support Table

	WWAN	WiFi	GPS	SIM0	SIM1
TREK-520	o	o	o	X	X
TREK-570	o	o	o	o	o
TREK-572	o	o	o	X	X
TREK-674	o	o	o	o	o

5.6.3 Enumeration

5.6.3.1 Windows/Linux

PERIPHERAL_TYPE enum

- (0) **PERIPHERAL_WWAN**
- (1) **PERIPHERAL_WIFI**
- (2) **PERIPHERAL_GPS**
- (3) **PERIPHERAL_ALTIMETER**
- (4) **PERIPHERAL_GSENSOR**
- (5) **PERIPHERAL_GPS_USB**
- (6) **PERIPHERAL_SPEAKER**
- (7) **PERIPHERAL_SIM0**
- (8) **PERIPHERAL_SIM1**
- (9) **PERIPHERAL_CAN**

SUSI_REAR_VIEW_SRC

- (0) SUSI_REAR_VIEW_SRC_SYSTEM (LVDS)
- (1) SUSI_REAR_VIEW_SRC_EXTERNAL1 (CVBS Camera Input)

5.6.3.2 Android

PERIPHERAL_TYPE enum

class:

mrm.define.IMC_ENUM.**PERIPHERAL_TYPE**

Enum:

Name	Type	value	Comment
PERIPHERAL_WWAN	int	0	
PERIPHERAL_WIFI	int	1	
PERIPHERAL_GPS	int	2	
PERIPHERAL_ALTIMETER	int	3	
PERIPHERAL_GSENSOR	int	4	
PERIPHERAL_GPS_USB	int	5	
PERIPHERAL_SPEAKER	int	6	
PERIPHERAL_SIM0	int	7	
PERIPHERAL_SIM1	int	8	
PERIPHERAL_CAN	int	9	

Remark:

Use the method **getValue()** to get the enum value.

ex: MRM_ENUM.PERIPHERAL_TYPE.PERIPHERAL_WIFI**.getValue()** returns 1.

Please refer to sample code for detailed usage.

SUSI_REAR_VIEW_SRC enum

class:

mrm.define.IMC_ENUM.**SUSI_REAR_VIEW_SRC**

Enum:

Name	Type	value	Comment
SUSI_REAR_VIEW_SRC_SYSTEM	int	0	LVDS
SUSI_REAR_VIEW_SRC_EXTERNAL1	int	1	CVBS Camera Input
SUSI_REAR_VIEW_SRC_EXTERNAL2	int	2	Reserved

Remark:

Use the method **getValue()** to get the enum value.

ex: MRM_ENUM.SUSI_REAR_VIEW_SRC.SUSI_REAR_VIEW_SRC_EXTERNAL1**.getValue()** returns 1.

Please refer to sample code for detailed usage.

5.6.4 APIs

5.6.4.1 SUSI_IMC_PERIPHERALCTRL_Initialize

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_PERIPHERALCTRL_Initialize**();

Description:

Initialize the peripheral control library.

Parameters:

None

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

- It shown the UART mapping table as following:

Hardware Platform	OS Platform	Port ID/PATH
TREK-520	WinCE	6
TREK-520	Linux	/proc/vpm_serial
TREK-674 TREK-572	Windows	1
TREK-674 TREK-572	Linux	/dev/ttyS0
TREK-570	Windows	4
TREK-570	Linux	/dev/ttyS3

Note: Linux ignore the port_number argument and using the variable PERIPHERAL_CTRL_XXXX_PATH which defined in /etc/susi/susi_config

5.6.4.2 SUSI_IMC_PERIPHERALCTRL_Deinitialize

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_PERIPHERALCTRL_Deinitialize**();

Description:

Deinitialize the peripheral control library.

Parameters:

None

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.6.4.3 SUSI_IMC_PERIPHERALCTRL_GetLibVersion

Syntax:

CE / Windows	USHORT **SUSI_IMC_PERIPHERALCTRL_GetLibVersion**(OUT BYTE version[18]);

Description:

Get the library version from TREK SDK.

Parameters:

pVersion [out]

Point to read buffer that specifies the peripheral control library's current version.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 CE

TREK-570 Windows

TREK-572 Windows

TREK-674 Windows

Remark:

The version buffer size must be greater than 18 bytes.

5.6.4.4 SUSI_IMC_PERIPHERALCTRL_SetPeripheralControl

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_PERIPHERALCTRL_SetPeripheralControl**(IN PERIPHERAL_TYPE nType, IN BOOL bEnable);
Android	short **SUSI_IMC_PERIPHERALCTRL_SetPeripheralControl**(int type, boolean enable)

Description:

Control peripheral status.

Parameters:

nType [in]

Linux / Windows

Peripheral type.

Android:

Peripheral type. Available value is enumerated in class PERIPHERAL_TYPE

bEnable [in]

Linux / Windows

Peripheral status.

Android:

Peripheral status.

TRUE for enable.

FALSE for disable.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.6.4.5 SUSI_IMC_PERIPHERALCTRL_GetPeripheralControl

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_PERIPHERALCTRL_GetPeripheralControl**(IN PERIPHERAL_TYPE nType, OUT BOOL& bEnable);
Android	short **SUSI_IMC_PERIPHERALCTRL_GetPeripheralControl**(int type, boolean[] enable)

Description:

Get peripheral status.

Parameters:

nType [in]

Linux / Windows

Peripheral type.

Android:

Peripheral type. Available value is enumerated in class PERIPHERAL_TYPE

bEnable [out]

Linux / Windows

Peripheral status.

Android

An allocated array of size 1 for storing the returned value of peripheral status. .

The return value will be stored at index 0.

TRUE for enable.

FALSE for disable.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.6.4.6 SUSI_IMC_PERIPHERALCTRL_SetRearViewSource

Syntax:

Linux / Windows	USHORT **SUSI_IMC_PERIPHERALCTRL_SetRearViewSource** (IN BYTE source);
Android	short **SUSI_IMC_PERIPHERALCTRL_SetRearViewSource**(byte source)

Description:

Set display source.

Parameters:

source [in]

Linux / Windows / Android:

Rear view source

SUSI_REAR_VIEW_SRC_SYSTEM=0 (LVDS)

SUSI_REAR_VIEW_SRC_EXTERNAL1=1 (CVBS Camera Input)

Android:

Rear view source

Available value is enumerated in class SUSI_REAR_VIEW_SRC

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows / Android

Remark:

Default is SUSI_REAR_VIEW_SRC_SYSTEM

5.6.4.7 SUSI_IMC_PERIPHERALCTRL_GetRearViewSource

Syntax:

Linux / Windows	USHORT **SUSI_IMC_PERIPHERALCTRL_GetRearViewSource** (OUT PBYTE source);
Android	short **SUSI_IMC_PERIPHERALCTRL_GetRearViewSource**(byte[] source)

Description:

Get display source status.

Parameters:

source [out]

Linux / Windows:

Display source.

SUSI_REAR_VIEW_SRC_SYSTEM=0

SUSI_REAR_VIEW_SRC_EXTERNAL1=1

Android:

An allocated array of size 1 for storing the returned value of display .

The return value will be stored at index 0.

Available value is enumerated in class SUSI_REAR_VIEW_SRC

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows / Android

Remark:

None

5.6.4.8 SUSI_IMC_PERIPHERALCTRL_SetAutoRearView

Syntax:

Linux / Windows	USHORT **SUSI_IMC_PERIPHERALCTRL_SetAutoRearView**(IN BOOL bEnable);
Android	short **SUSI_IMC_PERIPHERALCTRL_SetAutoRearView**(boolean enable)

Description:

Setup auto detect reserve gear swap display function.

Parameters:

bEnable [in]

Linux / Windows

Enable or Disable auto detect function

Android:

Enable/Disable Alarm status.

TRUE for enable.

FALSE for disable.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows / Android

Remark:

Control the auto detect reserve gear swap display function depend on digital input 1. Default disable this feature.

5.6.4.9 SUSI_IMC_PERIPHERALCTRL_GetAutoRearView

Syntax:

Linux / Windows	USHORT **SUSI_IMC_PERIPHERALCTRL_GetAutoRearView** (OUT PBOOL bEnable);
Android	short **SUSI_IMC_PERIPHERALCTRL_GetAutoRearView**(boolean[] enable)

Description:

Setup auto detect reserve gear swap display function.

Parameters:

bEnable [out]

Linux / Windows

Auto detect function status

Android

An allocated array of size 1 for storing the returned value of auto detect function status.

The return value will be stored at index 0.

TRUE for enable.

FALSE for disable.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows / Android

Remark:

None

5.7 GSENSOR

5.7.1 Usage Overview

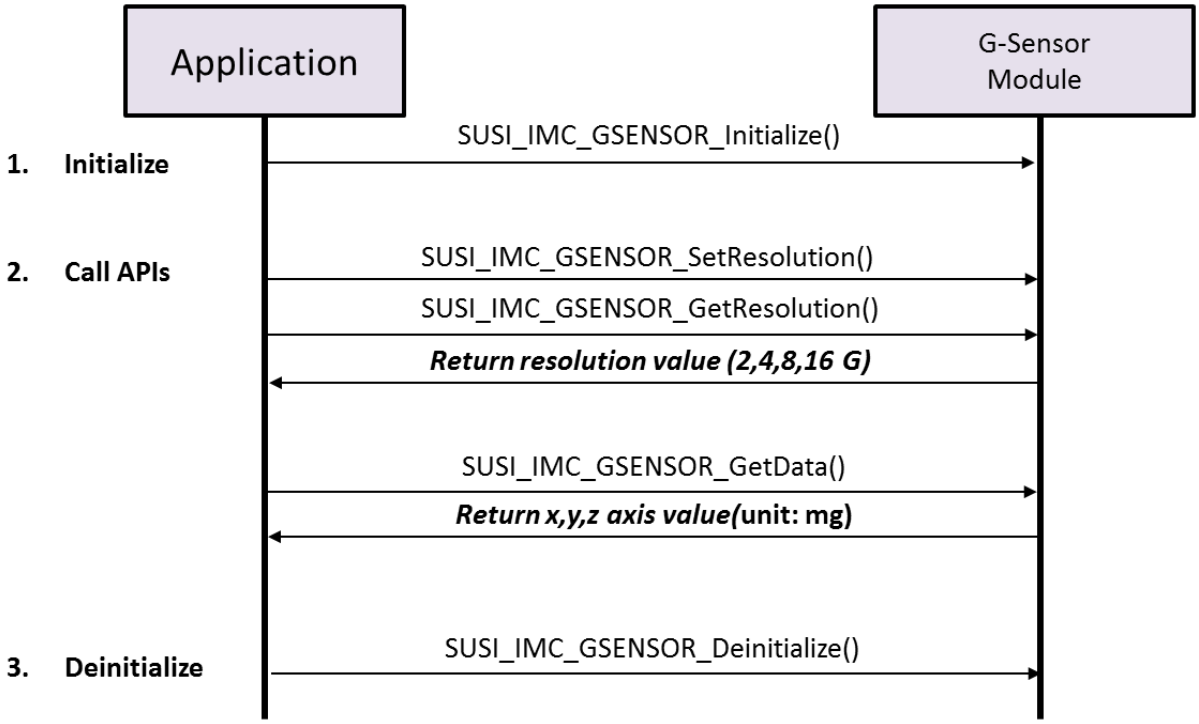
G-Sensor module provide API for controlling the G-sensor. For the basic usage, you can use the APIs to set the resolution of G-sensor and get the sensor data.

The following parts describe the API usage.

5.7.1.1 Basic Usage

This section describe the basic usage.

Windows/Linux



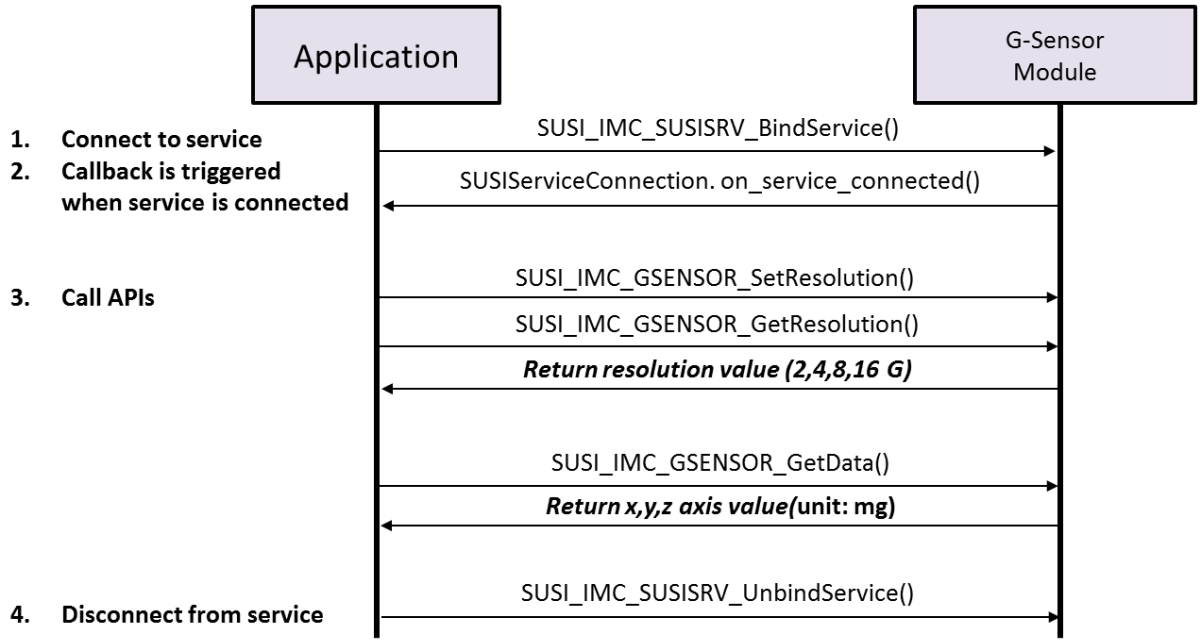
To start using G-Sensor module, you should first call SUSI_IMC_GSENSOR_Initialize() to initialize the G-sensor library and then you can start using APIs.

To set sensor resolution, call SUSI_IMC_GSENSOR_SetResolution() to set G-sensor (2,4,8,16G) resolution and call SUSI_IMC_GSENSOR_GetResolution() to get G-sensor resolution value.

To get sensor data, call SUSI_IMC_GSENSOR_GetData() to get current G-Sensor data of each(x,y,z) axis value.

To stop the G-Sensor module, use SUSI_IMC_GSENSOR_Deinitialize() to close API.

Android



The class for G sensor APIs for Android is **mrm.client.SUSIServiceClient**.

For the usage, please also refer to the section SUSI SRV Usage Overview.

After the SUSI service is connected, you can start using the APIs.

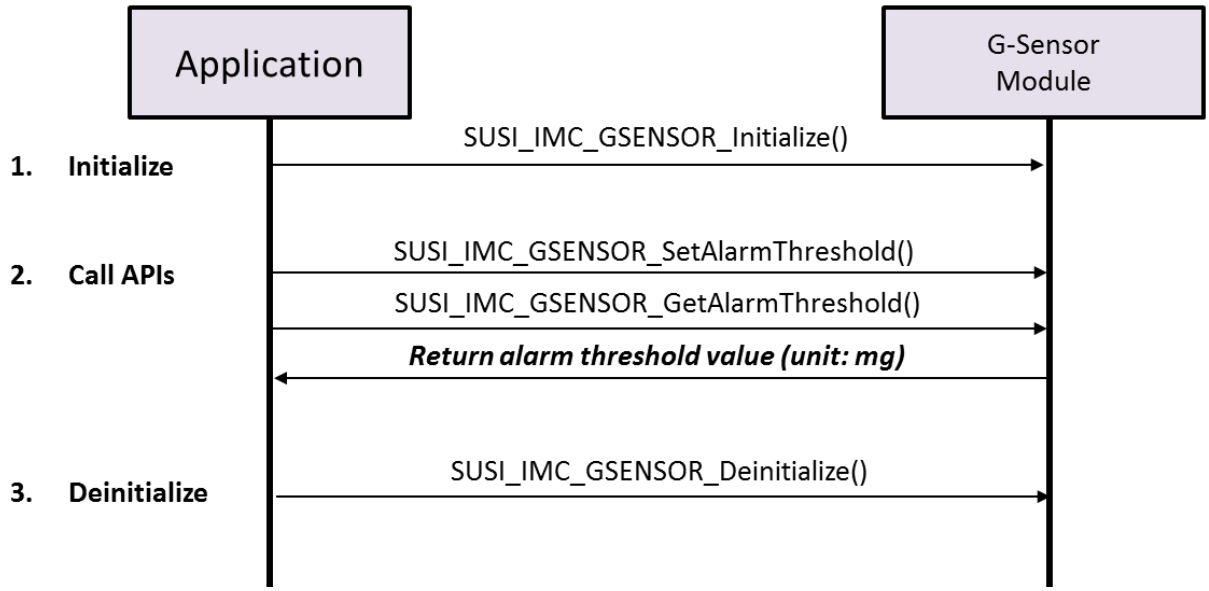
To set sensor resolution, call SUSI_IMC_GSENSOR_SetResolution() to set G-sensor (2,4,8,16G) resolution and call SUSI_IMC_GSENSOR_GetResolution() to get G-sensor resolution value.

To get sensor data, call SUSI_IMC_GSENSOR_GetData() to get current G-Sensor data of each(x,y,z) axis value.

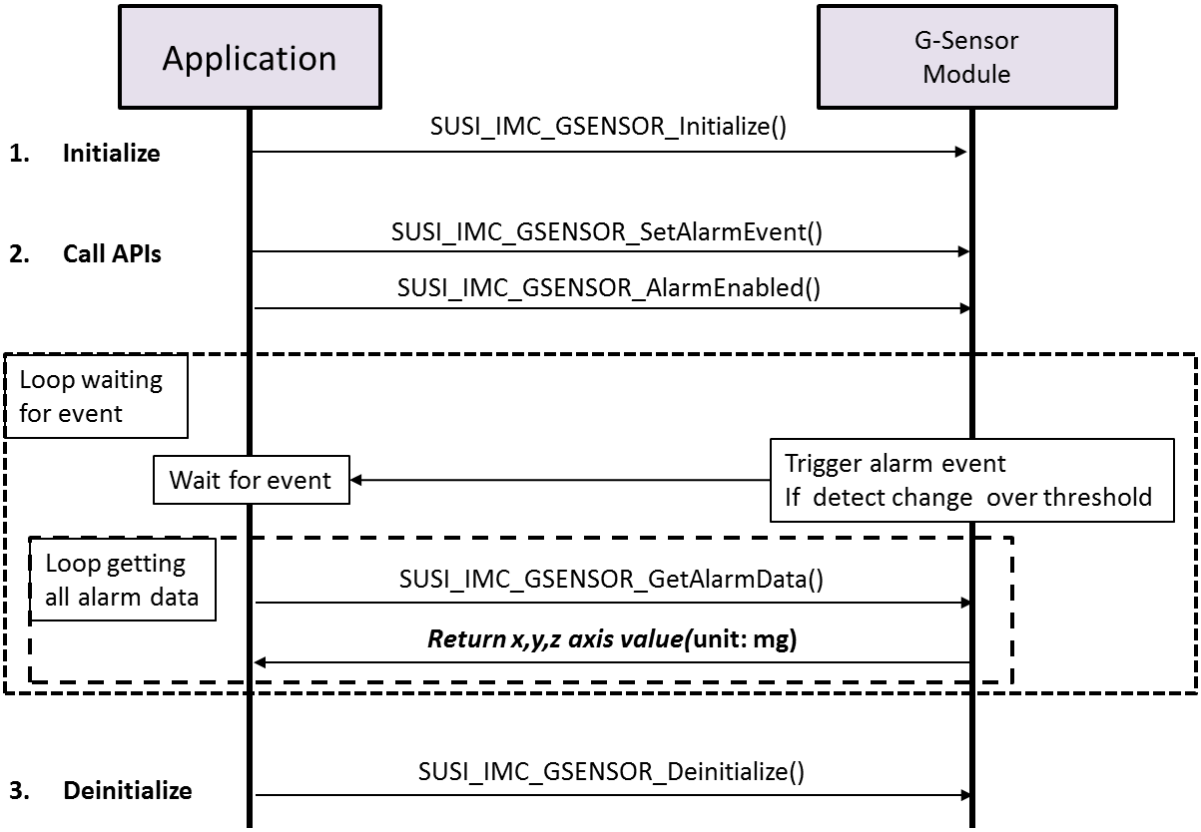
5.7.1.2 G Sensor Alarm Event

VPM is able to trigger alarm event when the G-Sensor sense dramatic change of the sensor data. The following parts describe how to set the alarm threshold, set the alarm event and get alarm data for each platform.

Windows/Linux



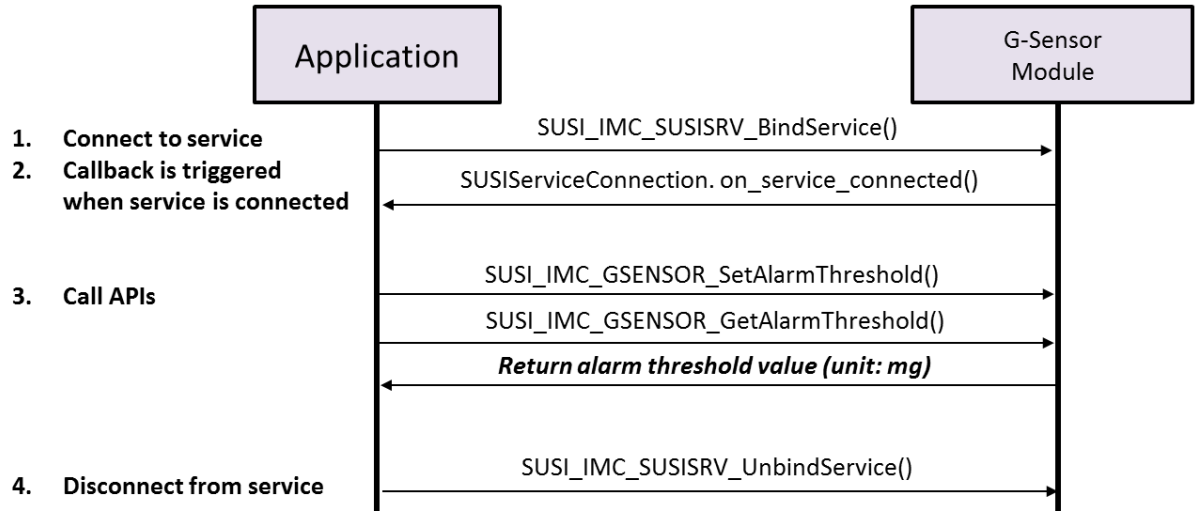
You can use SUSI_IMC_GSENSOR_SetAlarmThreshold() to set alarm threshold and use SUSI_IMC_GSENSOR_GetAlarmThreshold() to check current alarm threshold value.



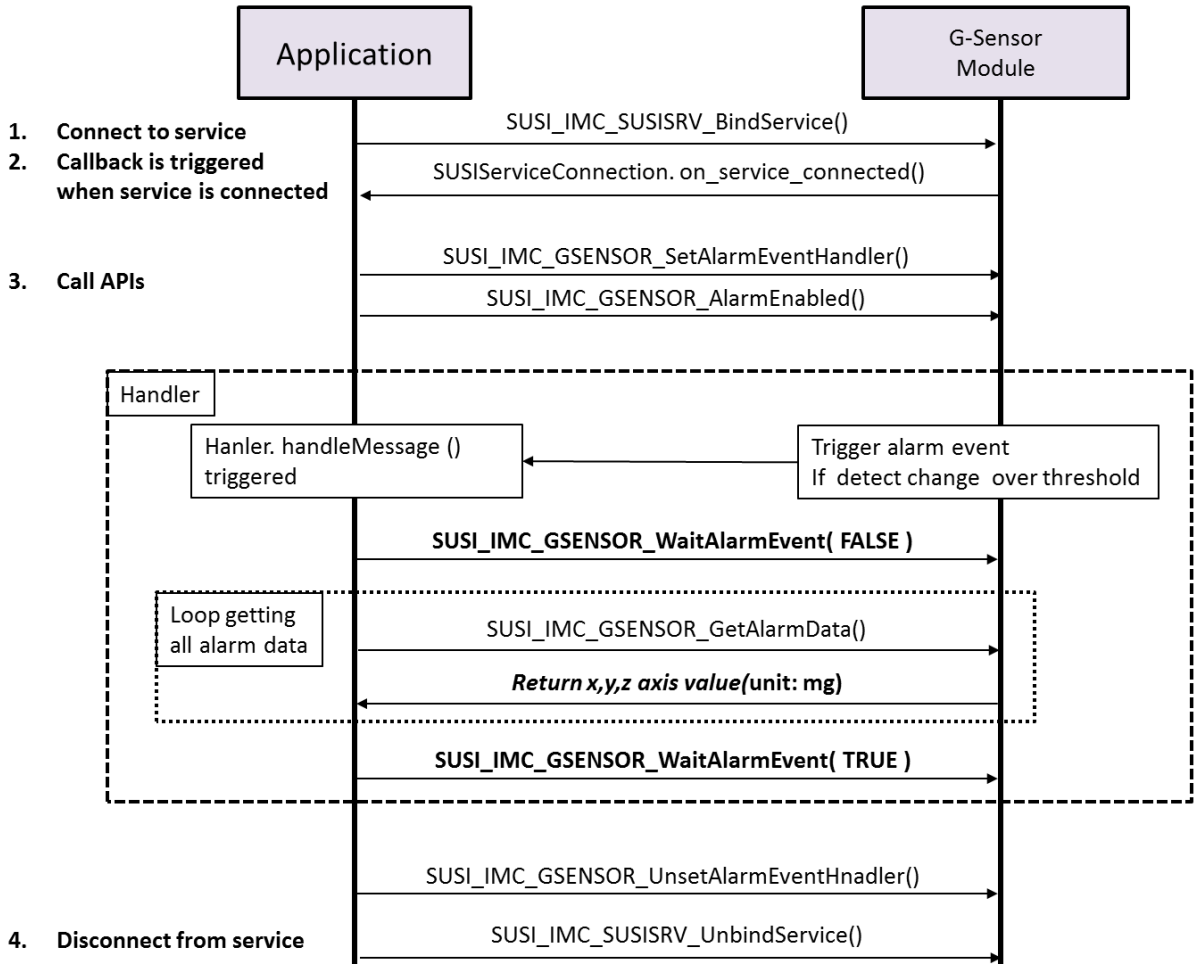
To enable the G sensor alarm event mechanism, you need to call SUSI_IMC_GSENSOR_SetAlarmEvent() to register an alarm event to G-Sensor API module. Then call SUSI_IMC_GSENSOR_AlarmEnabled() to enable alarm event function of VPM.
When G-sensor detect sensor data changes over specified threshold, the registered event will be triggered.

You need to hold an event listening loop in your APP to handle the registered event. SDK holds an internal buffer to store the alarm data. When the event is triggered, you can run loop to calling SUSI_IMC_GSENSOR_GetAlarmData() to consume the alarm data from the buffer.
After all alarm data are consumed, you should keep listening the registered event.
Please refer to the sample code for the details of implementation.

Android



You can use SUSI_IMC_GSENSOR_SetAlarmThreshold() to set alarm threshold and use SUSI_IMC_GSENSOR_GetAlarmThreshold() to check current alarm threshold value.



TREK SDK leverage the Android Handler mechanism to inform G sensor alarm event.

To enable the G sensor alarm event mechanism in Android, you need to create an instance of Handler and call SUSI_IMC_GSENSOR_SetAlarmEventHandler() to register the Handler instance to SUSI service. Then call SUSI_IMC_GSENSOR_AlarmEnabled() to enable alarm event function of VPM MCU.

When G-sensor detect sensor data changes over specified threshold, The registered event will be triggered and the handleMessage() callback of registered handler instance will be triggered.
SDK holds an internal buffer to store the alarm data. When the event is triggered, you can run loop to calling SUSI_IMC_GSENSOR_GetAlarmData() to consume the alarm data from the buffer.
Due to that SUSI service will trigger alarm event when it get a alarm data, to avoid from getting multiple unnecessary alarm events, you should call SUSI_IMC_GSENSOR_WaitAlarmEvent(FALSE) to ask SUSI service temporarily stop pass alarm event to handler before your APP start the alarm data getting loop. After your APP gets all alarm data, you should call SUSI_IMC_GSENSOR_WaitAlarmEvent(TRUE) to ask SUSI service continue the event informing.

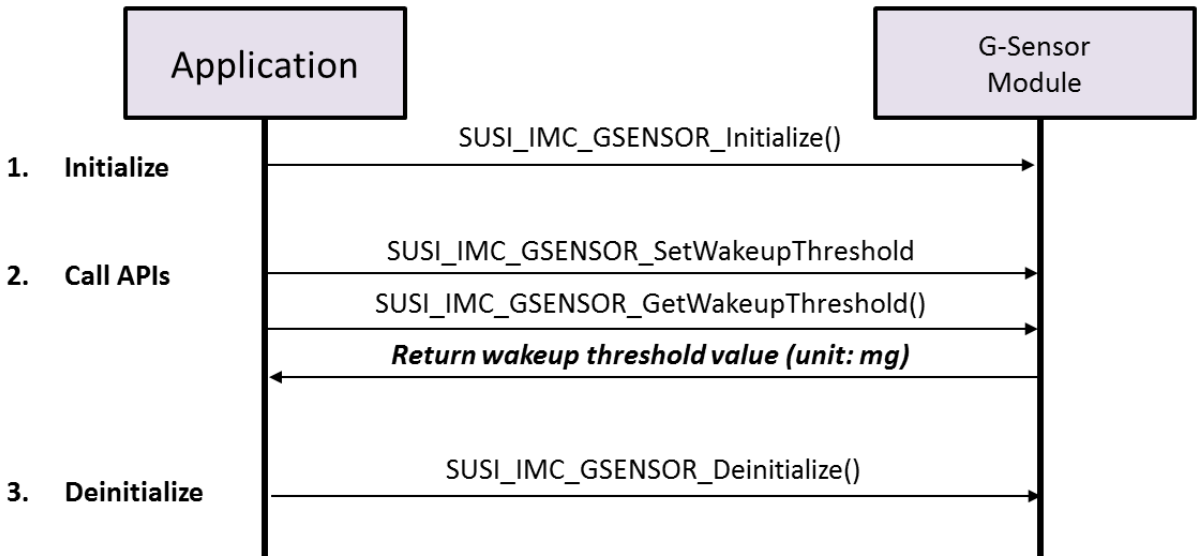
If you do not need to listen to alarm event anymore, you should call SUSI_IMC_GSENSOR_UnsetAlarmEventHandler() to unregister the Handler instance from SUSI service.

Please refer to the sample code for the details of implementation.

5.7.1.3 G Sensor Wakeup

VPM can set G-Sensor as wakeup source. The following parts describe how to configure the G sensor wakeup source.

Windows/Linux



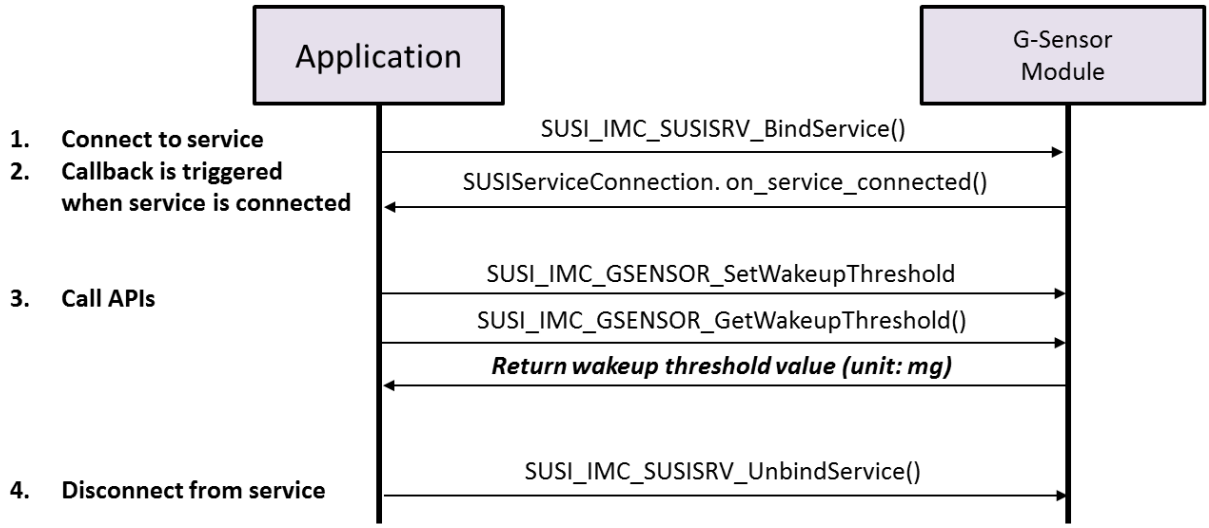
You can use SUSI_IMC_GSENSOR_SetWakeupThreshold() to set wakeup threshold and call

SUSI_IMC_GSENSOR_GetWakeupThreshold() to check current wakeup threshold value.

Also you should enable G sensor as wakeup source by using

SUSI_IMC_VPM_SetWakeupSourceControlStatus().

Android



You can use SUSI_IMC_GSENSOR_SetWakeupThreshold() to set wakeup threshold and call SUSI_IMC_GSENSOR_GetWakeupThreshold() to check current wakeup threshold value. Also you should enable G sensor as wakeup source by using SUSI_IMC_VPM_SetWakeupSourceControlStatus().

5.7.2 Enumeration

5.7.2.1 Windows/Linux

IMC_GSENSOR_RES

- (0) **PERIPHERAL_WWAN**
- (1) **PERIPHERAL_WIFI**
- (2) **PERIPHERAL_GPS**
- (3) **PERIPHERAL_ALTIMETER**
- (4) **PERIPHERAL_GSENSOR**
- (5) **PERIPHERAL_GPS_USB**
- (6) **PERIPHERAL_SPEAKER**
- (7) **PERIPHERAL_SIM0**
- (8) **PERIPHERAL_SIM1**
- (9) **PERIPHERAL_CAN**

5.7.2.2 Android

- (0) IMC_GSENSOR_RES_2G
- (1) IMC_GSENSOR_RES_4G
- (2) IMC_GSENSOR_RES_8G
- (3) IMC_GSENSOR_RES_16G

IMC_GSENSOR_RES enum

class:

mrm.define.IMC_ENUM.**IMC_GSENSOR_RES**

Enum:

Name	Type	value	Comment
IMC_GSENSOR_RES_2G	int	0	
IMC_GSENSOR_RES_4G	int	1	
IMC_GSENSOR_RES_8G	int	2	
IMC_GSENSOR_RES_16G	int	3	

Remark:

Use the method **getValue()** to get the enum value.

ex: MRM_ENUM.IMC_GSENSOR_RES.IMC_GSENSOR_RES_4G**.getValue()** returns 1.

Please refer to sample code for detailed usage.

5.7.3 Constant

5.7.3.1 Android

Class:
mrm.define.**IMC_CONSTANT**

Fields:

Field Name	Type	Value
SUSI_EVENT_ID_UNKNOWN	int	-1
SUSI_EVENT_ID_GSENSOR_ALARM	int	0

5.7.4 Structure/Classes

5.7.4.1 Windows/Linux

IMC_GSENSOR_AXIS_INFO Structure

Syntax:

```
typedef struct
{
        int x_mg;
        int y_mg;
        int z_mg;
}IMC_GSENSOR_DATA, *PIMC_GSENSOR_DATA;
```

Description:

This data structure defines the each 3-axis static acceleration of gravity value of G-Sensor.

Members:

x_mg

The x-axis acceleration of gravity value. The unit is mg.

y_mg

The y-axis acceleration of gravity value. The unit is mg.

z_mg

The z-axis acceleration of gravity value. The unit is mg.

5.7.4.2 Android

IMC_GSENSOR_DATA

class:

mrm.define.gsensor.**IMC_GSENSOR_DATA**

Fields:

Field Name	Type	Input/Output	Comment
x_mg	int	in, out	
y_mg	int	in, out	
z_mg	int	in, out	

5.7.5 APIs

5.7.5.1 SUSI_IMC_GSENSOR_Initialize

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_GSENSOR_Initialize**(IN BYTE byPortNo);

Description:

Initialize the gsensor control library.

Parameters:

None

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

- It shown the UART mapping table as following:

Hardware Platform	OS Platform	Port ID/PATH
TREK-520	WinCE	6
TREK-520	Linux	/proc/vpm_serial
TREK-674 TREK-572	Windows	1
TREK-674 TREK-572	Linux	/dev/ttyS0
TREK-570	Windows	4
TREK-570	Linux	/dev/ttyS3

Note: Linux ignore the port_number argument and using the variable GSN_PATH which defined in /etc/susi/susi_config

5.7.5.2 SUSI_IMC_GSENSOR_Deinitialize

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_GSENSOR_Deinitialize();**

Description:

Deinitialize the gsensor library.

Parameters:

None

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.7.5.3 SUSI_IMC_GSENSOR_GetLibVersion

Syntax:

CE / Windows	USHORT **SUSI_IMC_GSENSOR_GetLibVersion**(OUT BYTE version[18]);

Description:

Get the library version from TREK SDK.

Parameters:

pVersion [out]

Point to read buffer that specifies the gsensor library's current version.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 CE

TREK-570 Windows

TREK-572 Windows

TREK-674 Windows

Remark:

The version buffer size must be greater than 18 bytes.

5.7.5.4 SUSI_IMC_GSENSOR_GetResolution

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_GSENSOR_GetResolution**(OUT IMC_GSENSOR_RES *res);
Android	short **SUSI_IMC_GSENSOR_GetResolution**(int[] res)

Description:

Get the G-sensor resolution value.

Parameters:

res [out]

Linux / Windows

G sensor resolution value:

0: IMC_GSENSOR_RES_2G

1: IMC_GSENSOR_RES_4G

2: IMC_GSENSOR_RES_8G

3: IMC_GSENSOR_RES_16G

Android

An allocated array of size 1 for storing the returned value of G sensor resolution .

The return value will be stored at index 0.

Available value is enumerated in class IMC_GSENSOR_RES

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.7.5.5 SUSI_IMC_GSENSOR_SetResolution

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_GSENSOR_SetResolution**(IN IMC_GSENSOR_RES res);
Android	short **SUSI_IMC_GSENSOR_SetResolution**(int res)

Description:

Set the G-sensor resolution value.

Parameters:

res [in]

Linux / Windows

G sensor resolution value:

0: IMC_GSENSOR_RES_2G

1: IMC_GSENSOR_RES_4G

2: IMC_GSENSOR_RES_8G

3: IMC_GSENSOR_RES_16G

Android

G sensor resolution value:

Available values are enumerated in class IMC_GSENSOR_RES

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.7.5.6 SUSI_IMC_GSENSOR_GetData

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_GSENSOR_GetData**(OUT PIMC_GSENSOR_DATA data);
Android	short **SUSI_IMC_GSENSOR_GetData**(IMC_GSENSOR_DATA data)

Description:

This API will get G-Sensor each axis value.

Parameters:

data [out]

Linux / Windows

Point to read buffer that specifies the data of each axis.

Android

A instance of class **IMC_GSENSOR_DATA** for storing the returned values of data of each axis.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.7.5.7 SUSI_IMC_GSENSOR_GetAlarmThreshold

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_GSENSOR_GetAlarmThreshold**(OUT int *mg);
Android	short **SUSI_IMC_GSENSOR_GetAlarmThreshold**(int[] mg)

Description:

Get the G-sensor alarm threshold.

Parameters:

mg [out]

G-sensor alarm threshold. If the gravity difference bigger than this threshold; the alarm event will trigger when the alarm function is turn on.

The unit is mg. The range is 2000~15000 mg. The scale factor is 62.5 mg.

Linux / Windows

Pointer to variable storing returned value.

Android:

An allocated array of size 1 for storing the returned value of alarm threshold.

The return value will be stored at index 0.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android, firmware version >= 1.020

TREK-674 Linux / Windows , firmware version >= 1.014

TREK-572 Linux / Windows

Remark:

None

5.7.5.8 SUSI_IMC_GSENSOR_SetAlarmThreshold

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_GSENSOR_SetAlarmThreshold**(IN int mg);
Android	short **SUSI_IMC_GSENSOR_SetAlarmThreshold**(int mg)

Description:

Set the G-sensor alarm threshold.

Parameters:

mg [in]

G-sensor alarm threshold. If the gravity difference bigger than this threshold; the alarm event will trigger when the alarm function is turn on.

The unit is mg. The range is 2000~15000 mg. The scale factor is 62.5 mg.

Linux / Windows

The threshold value

Android:

The threshold value

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android, firmware version >= 1.020

TREK-674 Linux / Windows , firmware version >= 1.014

TREK-572 Linux / Windows

Remark:

None

5.7.5.9 SUSI_IMC_GSENSOR_SetAlarmEvent

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_GSENSOR_SetAlarmEvent**(IN void* Event);

Description:

Set alarm event to listen G-sensor alarm.

Parameters:

hEvent [in]

Event handler of G-sensor alarm event.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows , firmware version >= 1.020

TREK-674 Linux / Windows , firmware version >= 1.014

TREK-572 Linux / Windows

Remark:

None

5.7.5.10 SUSI_IMC_GSENSOR_SetAlarmEventHandler

Syntax:

Android	short **SUSI_IMC_GSENSOR_SetAlarmEventHandler**(Handler handler)

Description:

Set handler which handles G-sensor alarm event.

Parameters:

handler [in]

An instance of Handler.. Can not be null.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Android, firmware version >= 1.020

Remark:

- Please refer to the usage guide and sample code for for details.
- On alarm event triggered, the Handler will receive message with the "what" field equals to SUSI_EVENT_ID_GSENSOR_ALARM

5.7.5.11 SUSI_IMC_GSENSOR_UnsetAlarmEventHandler

Syntax:

Android	short **SUSI_IMC_GSENSOR_UnsetAlarmEventHandler**()

Description:

Unregister handler of G-sensor alarm event.

Parameters:

none

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Android, firmware version >= 1.020

Remark:

- Please refer to the usage guide and sample for for details.

5.7.5.12 SUSI_IMC_GSENSOR_WaitAlarmEvent

Syntax:

Android	short **SUSI_IMC_GSENSOR_WaitAlarmEvent**(boolean status)

Description:

Allow/Disallow SUSI service to pass G-sensor alarm event to registered handler when VPM triggers alarm event.

Parameters:

status [in]

The status of whether SUSI service should pass G sensor alarm event to registered handler

TRUE: Inform

FLASE: Not to inform

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Android, firmware version >= 1.020

Remark:

- Please refer to the usage guide and sample for for details.

5.7.5.13 SUSI_IMC_GSENSOR_AlarmEnabled

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_GSENSOR_AlarmEnabled**(bool bActive);
Android	short **SUSI_IMC_GSENSOR_AlarmEnabled**(boolean active)

Description:

Set VPM G-sensor alarm mode status.

Parameters:

bActive [in]

Linux / Windows

Status value

TRUE : Enable G-sensor alarm mode.

FALSE : Disable G-sensor alarm mode.

Android:

Status value

TRUE : Enable G-sensor alarm mode.

FALSE : Disable G-sensor alarm mode.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android, firmware version >= 1.020

TREK-674 Linux / Windows , firmware version >= 1.014

TREK-572 Linux / Windows

Remark:

None

5.7.5.14 SUSI_IMC_GSENSOR_GetAlarmData

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_GSENSOR_GetAlarmData** (OUT PIMC_GSENSOR_DATA data);
Android	short **SUSI_IMC_GSENSOR_GetAlarmData**(IMC_GSENSOR_DATA data)

Description:

This API will get G-Sensor Alarm trigger data value each axis value from library queue. The library has a queue structure to save the alarm trigger data until user to calling this function to fetch data; otherwise the queue's data will be overwritten.

Parameters:

data [out]

Linux / Windows

Point to read buffer that specifies the data of each axis.

Android

A instance of class **IMC_GSENSOR_DATA** for storing the returned values of data of each axis.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android, firmware version >= 1.020

TREK-674 Linux / Windows , firmware version >= 1.014

TREK-572 Linux / Windows

Remark:

If should calling this function until the return IMC_ERR_GSENSOR_DATA_NOT_READY

5.7.5.15 SUSI_IMC_GSENSOR_GetWakeupThreshold

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_GSENSOR_GetWakeupThreshold**(OUT int *mg);
Android	short **SUSI_IMC_GSENSOR_GetWakeupThreshold**(int[] mg)

Description:

Get the G-sensor wakeup threshold.

Parameters:

mg [out]

G-sensor wakeup threshold. If the gravity difference bigger than this threshold; the platform will wakeup when the G sensor wakeup function turn on.

The unit is mg. The range is 63~16000 mg. The scale factor is 62.5 mg.

Linux / Windows

Pointer to variable storing returned value.

Android:

An allocated array of size 1 for storing the returned value of threshold.

The return value will be stored at index 0.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.7.5.16 SUSI_IMC_GSENSOR_SetWakeupThreshold

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_GSENSOR_SetWakeupThreshold**(IN int mg);
Android	short **SUSI_IMC_GSENSOR_SetWakeupThreshold**(int mg)

Description:

Set the G-sensor wakeup threshold.

Parameters:

mg [in]

G-sensor wakeup threshold. If the gravity difference bigger than this threshold; the platform will wakeup when the G sensor wakeup function turn on.

The unit is mg. The range is (63, 16000) mg. The scale factor is 62.5 mg.

Linux / Windows

The threshold value

Android:

The threshold value

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

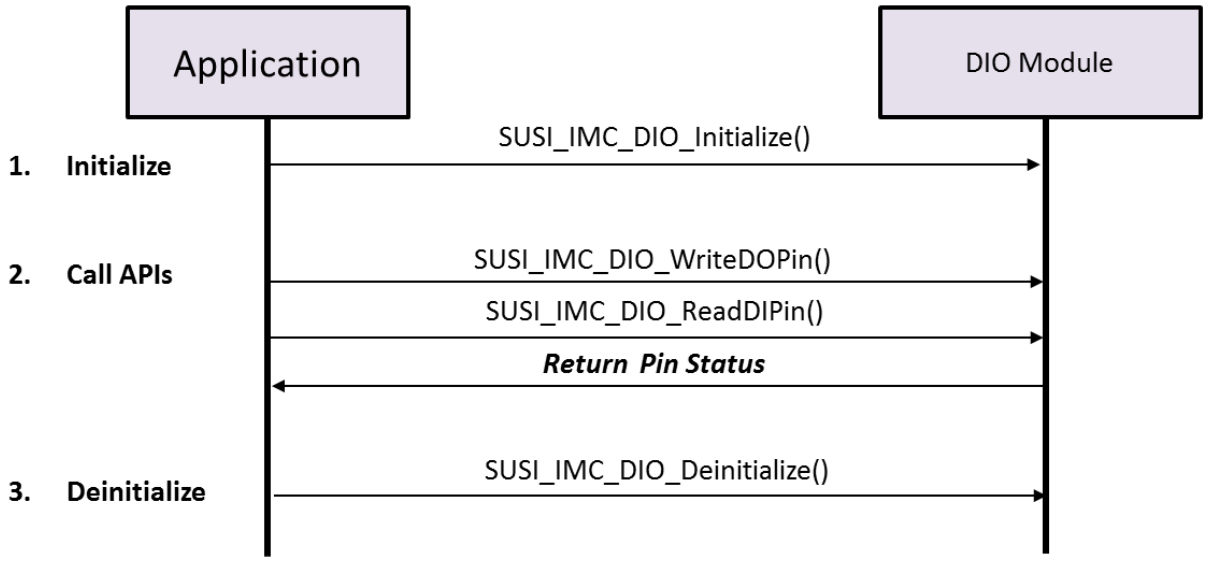
5.8 DIO

5.8.1 Usage Overview

TREK devices are equipped with Digital IO(DIO) pins which can be used to control external devices. The DIO API module provides APIs to set the digital output(DO) pins status and get the digital input(DI) pins status.

The following parts describe the API usage.

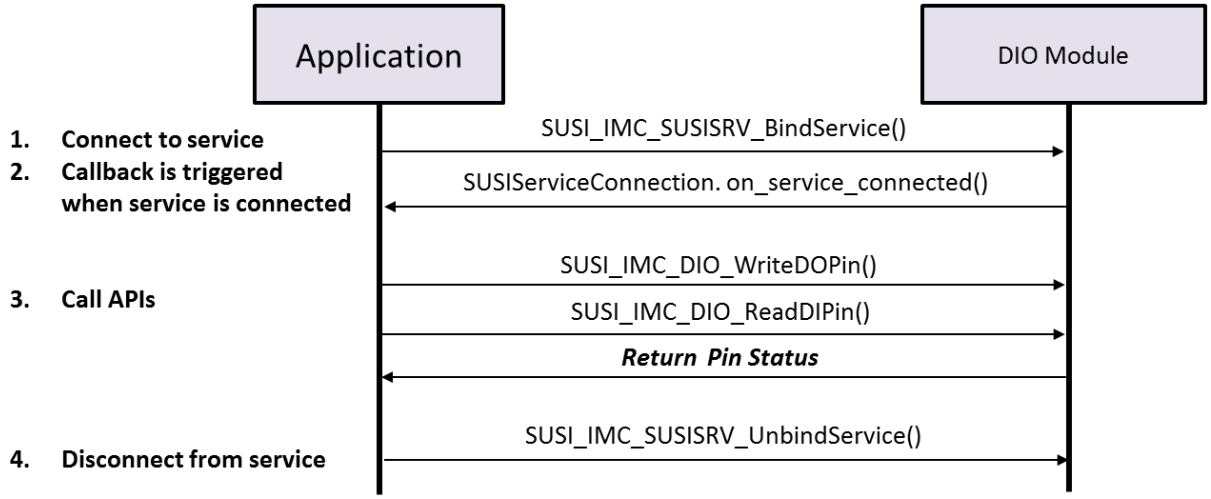
5.8.1.1 Windows



You need call SUSI_IMC_DIO_Initialize() to Initialize the DIO library. And then, use SUSI_IMC_DIO_ReadDIPin() to read DI pin status. Also can use SUSI_IMC_DIO_WriteDOPin() to enable/disable DO pin. To stop the DIO module, you must use SUSI_IMC_DIO_Deinitialize() to close API .

Remark: The Digital I/O need install I/O Port Driver on Windows 7/8. You can found the driver on ***<SDK>/driver***. Right-click on ***install.bat*** and select *Run as administrator* to install. If you success installed the driver, you should see ***Advantech Proprietary Devices*** on your device manager

5.8.1.2 Android



The class for G sensor APIs for Android is **mrm.client.SUSIServiceClient**.

For the usage, please also refer to the section SUSI SRV Usage Overview.

After the SUSI service is connected, you can start using the APIs. You cab use SUSI_IMC_DIO_ReadDIPin() to read DI pin status and use SUSI_IMC_DIO_WriteDOPin() to enable/disable DO pin.

5.8.2 Structure/Classes

5.8.2.1 Windows/Linux

PIN_STATUS

Syntax:

```
struct PIN_STATUS
{
        BOOL bPin0;
        BOOL bPin1;
        BOOL bPin2;
        BOOL bPin3;
};
typedef PIN_STATUS* PPIN_STATUS;
```

Description:

This data structure defines the status of each Digital Input/Output pins to be get/set.

Members:

bPin0

Status of Digital Input/Output pin 1.

bPin1

Status of Digital Input/Output pin 2.

bPin2

Status of Digital Input/Output pin 3.

bPin3

Status of Digital Input/Output pin 4.

5.8.2.2 Android

PIN_STATUS

class:

mrm.define.dio.**PIN_STATUS**

Fields:

Field Name	Type	Input/Output	Comment
bPin0	boolean	in, out	Status of Digital Input/Output pin 1.
bPin1	boolean	in, out	Status of Digital Input/Output pin 2.
bPin2	boolean	in, out	Status of Digital Input/Output pin 3.
bPin3	boolean	in, out	Status of Digital Input/Output pin 4.

5.8.3 APIs

5.8.3.1 SUSI_IMC_DIO_Initialize

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_DIO_Initialize**(void);

Description:

Initialize the DIO library.

Parameters:

None

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.8.3.2 SUSI_IMC_DIO_Deinitialize

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_DIO_Deinitialize**(void);

Description:

Deinitialize the DIO library.

Parameters:

None

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows

TREK-674 Linux / Windows

Remark:

It's required to call this function to release the WIFI library to avoid the memory leak.

5.8.3.3 SUSI_IMC_DIO_GetLibVersion

Syntax:

CE / Windows	USHORT **SUSI_IMC_DIO_GetLibVersion**(OUT BYTE version[18]);

Description:

Get the library version from DIO library.

Parameters:

pVersion [out]

Point to read buffer that specifies the DIO library's current version.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 CE

TREK-570 Windows

TREK-674 Windows

Remark:

The version buffer size must be greater than 18 bytes.

5.8.3.4 SUSI_IMC_DIO_ReadDIPin

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_DIO_ReadDIPin**(PIN_STATUS& PinStatus);
Android	short **SUSI_IMC_DIO_ReadDIPin**(PIN_STATUS pinStatus)

Description:

Read DI pin.

Parameters:

PinStatus [out]

Linux / Windows

The pin status.

Android

A instance of class **PIN_STATUS** for storing the status of all DI pins.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-674 Linux / Windows

Remark:

None

5.8.3.5 SUSI_IMC_DIO_WriteDOPin

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_DIO_WriteDOPin**(const PIN_STATUS& PinStatus);
Android	short **SUSI_IMC_DIO_WriteDOPin**(PIN_STATUS pinStatus)

Description:

Write DO pin.

Parameters:

PinStatus [in]

Linux / Windows

The pin status.

Android

A instance of class **PIN_STATUS** for storing the status of all DO pins to be set.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.8.3.6 SUSI_IMC_DIO_GetVoiceSourceControlPin

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_DIO_GetVoiceSourceControlPin**(OUT PINT PinStatus);
Android	short **SUSI_IMC_DIO_GetVoiceSourceControlPin**(int[] PinStatus)

Description:

Get Voice Source Control Pin status.

Parameters:

PinStatus [out]

The Voice Source Control pin status.

IMC_DIO_VOICE_SOURCE_BLUETOOTH (0)

IMC_DIO_VOICE_SOURCE_MIC_IN (1)

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.8.3.7 SUSI_IMC_DIO_SetVoiceSourceControlPin

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_DIO_SetVoiceSourceControlPin**(IN INT PinStatus);
Android	short **SUSI_IMC_DIO_SetVoiceSourceControlPin**(int PinStatus)

Description:

Set Voice Source Control Pin status.

Parameters:

PinStatus [in]

The Voice Source Control pin status.

IMC_DIO_VOICE_SOURCE_BLUETOOTH (0)

IMC_DIO_VOICE_SOURCE_MIC_IN (1)

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.8.3.8 SUSI_IMC_DIO_GetBluetoothEnablePin

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_DIO_GetBluetoothEnablePin**(OUT PINT PinStatus);
Android	short **SUSI_IMC_DIO_GetBluetoothEnablePin**(int[] PinStatus)

Description:

Get Bluetooth Enable Pin status.

Parameters:

PinStatus [out]

The Bluetooth Enable pin status.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows

Remark:

None

5.8.3.9 SUSI_IMC_DIO_SetBluetoothEnablePin

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_DIO_SetBluetoothEnablePin**(IN INT PinStatus);
Android	short **SUSI_IMC_DIO_SetBluetoothEnablePin**(int PinStatus)

Description:

Set Bluetooth Enable Control Pin status.

Parameters:

PinStatus [in]

The Bluetooth Enable pin status.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows

Remark:

None

5.8.3.10 SUSI_IMC_DIO_GetVoiceModuleSelectPin

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_DIO_GetVoiceModuleSelectPin** (OUT PINT PinStatus);
Android	short **SUSI_IMC_DIO_GetVoiceModuleSelectPin** (int[] PinStatus)

Description:

Get Voice Module Select Pin status.

Parameters:

PinStatus [out]

The Voice Module select pin status.

IMC_DIO_VOICE_MODULE_MX809X (0)

IMC_DIO_VOICE_MODULE_MX73X4 (1)

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-674 Linux / Windows

Remark:

None

5.8.3.11 SUSI_IMC_DIO_SetVoiceModuleSelectPin

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_DIO_SetVoiceModuleSelectPin** (IN INT PinStatus);
Android	short **SUSI_IMC_DIO_SetVoiceModuleSelectPin** (int PinStatus)

Description:

Set Voice Module Select Pin status.

Parameters:

PinStatus [in]

The Voice Module Select pin status.

IMC_DIO_VOICE_MODULE_MX809X (0)

IMC_DIO_VOICE_MODULE_MX73X4 (1)

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-674 Linux / Windows

Remark:

None

5.9 IIC

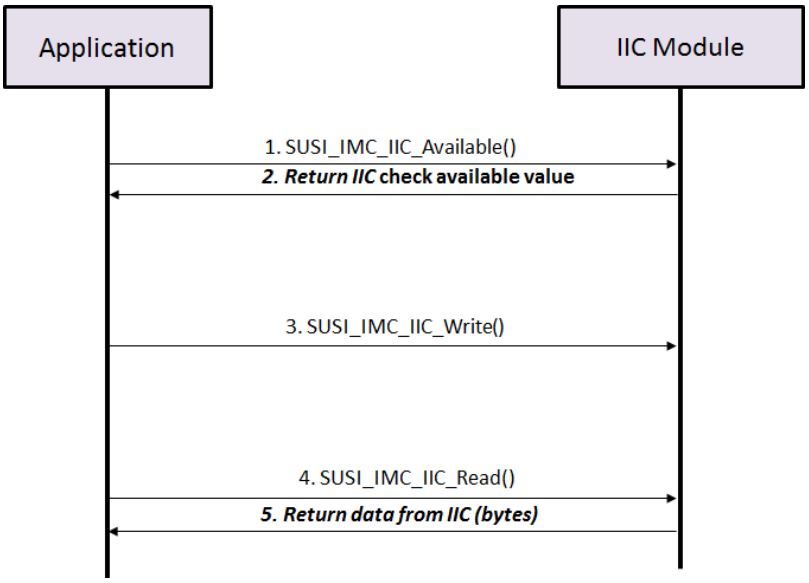
5.9.1 Usage Overview

IIC(I2C) module provide read/write data by I2C bus. The I2C API module provides APIs to access I2C bus.

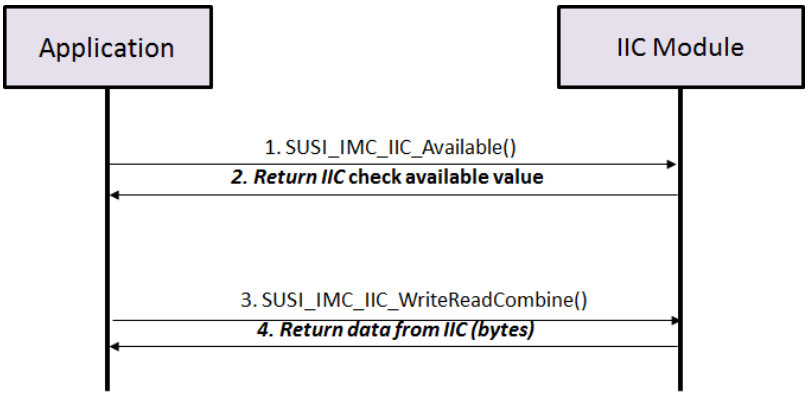
The following parts describe the API usage.

5.9.1.1 Windows/Linux

You need call SUSI_IMC_IIC_Available() to check I2C driver is available and also get the I2C type supported(SMBus or I2C). Then, call SUSI_IMC_IIC_Write() to write data(bytes) to the target slave address by the I2C bus and use SUSI_IMC_IIC_Read() to read data(bytes) from the target slave address by the I2C bus.



You also can use SUSI_IMC_IIC_WriteReadCombine() for a sequential operation to write data and then read data from the target slave address by the I2C bus.



5.9.2 Structure/Classes

5.9.2.1 Windows/Linux

IMC_IIC_READ Structure

Syntax:

```
typedef struct
{
        DWORD IICType;
        BYTE SlaveAddress;
        BYTE *ReadBuf;
        DWORD ReadLen;
}IMC_IIC_READ,*PIMC_IIC_READ;
```

Description:

This structure defines the I2C parameter for read .

Members:

IICType [in]

Specifies the I2C type, the value can either be

1 : Primary mode

2 : SMBus mode

SlaveAddress [in]

Specifies the 8-bit device address, ranging from 0x00 – 0xFF.

Whether to give a 1 (read) or 0 (write) to the LSB of SlaveAddress could be ignored.

ReadBuf [out]

Pointer to a variable in which the function reads the bytes of data.

ReadLen [in]

Specifies the number of bytes to be read.

IMC_IIC_WRITE Structure

Syntax:

```
typedef struct
{
        DWORD IICType;
        BYTE SlaveAddress;
        BYTE *WriteBuf;
        DWORD WriteLen;
}IMC_IIC_WRITE,*PIMC_IIC_WRITE;
```

Description:

This structure defines the I2C parameter for write.

Members:

IICType [in]

Specifies the I2C type, the value can either be

1 : Primary mode

2 : SMBus mode

SlaveAddress [in]

Specifies the 8-bit device address, ranging from 0x00 – 0xFF.

Whether to give a 1 (read) or 0 (write) to the LSB of SlaveAddress could be ignored.

WriteBuf [in]

Pointer to a byte array which contains the bytes of data to be written.

WriteLen [in]

Specifies the number of bytes to be written.

IMC_IIC_WRITE_READ_COMBINE Structure

Syntax:

```
typedef struct
{
        BYTE SlaveAddress;
        BYTE *WriteBuf;
        DWORD WriteLen;
        BYTE *ReadBuf;
        DWORD ReadLen;
}IMC_IIC_WRITE_READ_COMBINE,*PIMC_IIC_WRITE_READ_COMBINE;
```

Description:

This structure defines the I2C parameter for write and read of a sequential operation.

Members:

IICType [in]

Specifies the I2C type, the value can either be

1 : Primary mode

2 : SMBus mode

SlaveAddress [in]

Specifies the 8-bit device address, ranging from 0x00 – 0xFF.

Whether to give a 1 (read) or 0 (write) to the LSB of SlaveAddress could be ignored.

WriteBuf [in]

Pointer to a byte array which contains the bytes of data to be written.

WriteLen [in]

Specifies the number of bytes to be written.

ReadBuf [out]

Pointer to a variable in which the function reads the bytes of data.

ReadLen [in]

Specifies the number of bytes to be read.

5.9.3 APIs

5.9.3.1 SUSI_IMC_IIC_GetLibVersion

Syntax:

CE	USHORT **SUSI_IMC_IIC_GetLibVersion**(OUT BYTE version[18]);

Description:

Get current version from the IIC library.

Parameters:

version [out]

Point to read buffer that specifies the IIC library's current version.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 CE

Remark:

The version buffer size must be greater than 18 bytes.

5.9.3.2 SUSI_IMC_IIC_Available

Syntax:

Linux / CE	USHORT **SUSI_IMC_IIC_Available**(void);

Description:

Check if I2C driver is available and also get the IIC type supported.

Parameters:

None

Returns:

- Value range

  IMC_ERR_IIC_STATUS_OK_PLATFORM_NOT_SUPPORT_ALL - The function succeeds; the platform does not support any IIC - APIs.

  IMC_ERR_IIC_STATUS_OK_PLATFORM_ONLY_SUPPORT_PRIMARY - The function succeeds; the platform supports only primary IIC.

  IMC_ERR_IIC_STATUS_OK_PLATFORM_ONLY_SUPPORT_SMBUS - The function succeeds; the platform supports only SMBus implemented IIC.

  IMC_ERR_IIC_STATUS_OK_PLATFORM_SUPPORT - The function succeeds; the platform supports both primary IIC and SMBus IIC.
- Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

Remark:

None

5.9.3.3 SUSI_IMC_IIC_Read

Syntax:

Linux / CE	USHORT **SUSI_IMC_IIC_Read**(PIMC_IIC_READ parm);

Description:

Read bytes of data from the target slave device in the I2C bus.

Parameters:

parm [in/out]

Retrieve data form I2C bus.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

Remark:

None

5.9.3.4 SUSI_IMC_IIC_Write

Syntax:

Linux / CE	USHORT **SUSI_IMC_IIC_Write**(PIMC_IIC_WRITE parm);

Description:

Write bytes of data to the target slave device in the I2C bus.

Parameters:

parm [in/out]

Send data to I2C bus.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

Remark:

None

5.9.3.5 SUSI_IMC_IIC_WriteReadCombine

Syntax:

Linux / CE	USHORT **SUSI_IMC_IIC_WriteReadCombine**(PIMC_IIC_WRITE_READ_COMBINE parm);

Description:

A sequential operation to write bytes of data followed by bytes read from the target slave device in the I2C bus.

Parameters:

parm [in/out]

Send and read data on I2C bus.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

Remark:

None

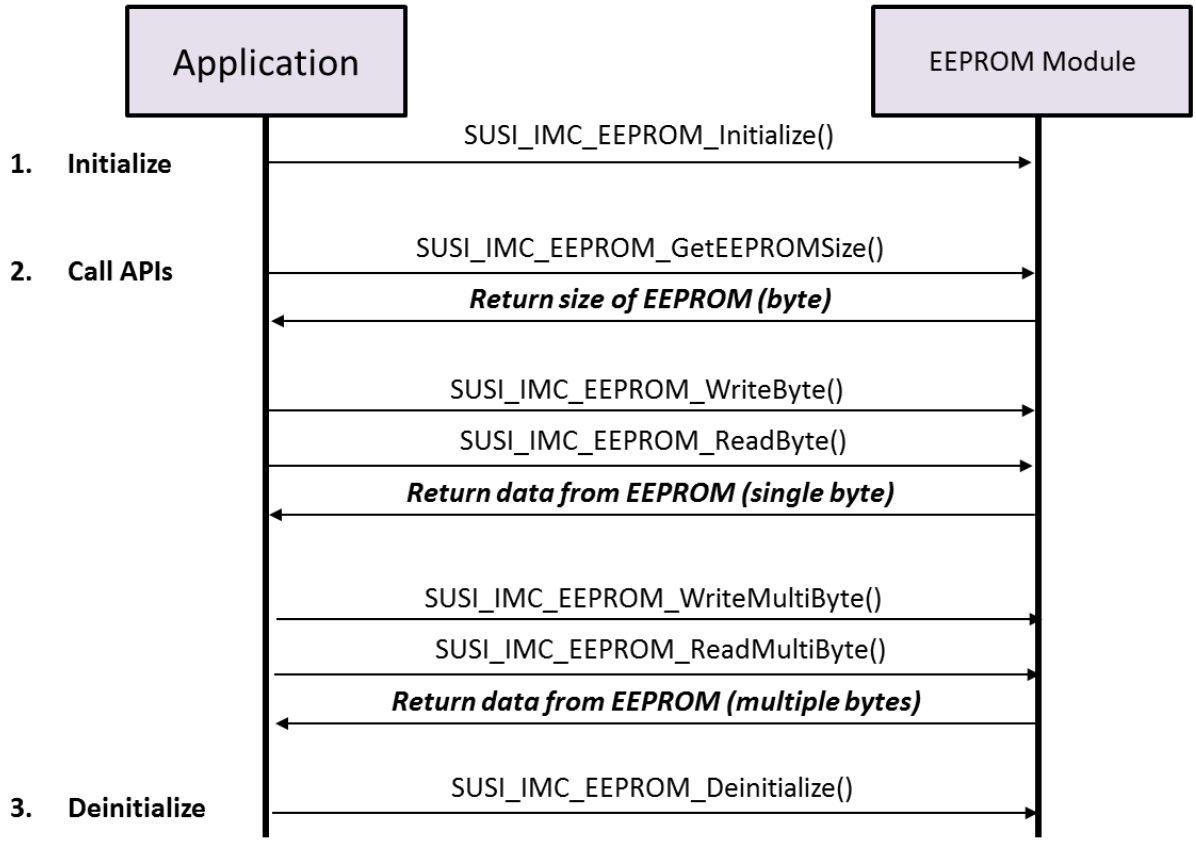
5.10 EEPROM

5.10.1 Usage Overview

EEPROM module provides APIs to save/load arbitrary data to the EEPROM on the TREK device.
The following parts describe the API usage.

5.10.1.1 Windows/Linux

In the Linux platform, you should ensure the i2c driver (i2c-dev) is loaded before calling this function. Using the following command to load the i2c driver (# modprobe i2c-dev).



To start using EEPROM APIs, you need call SUSI_IMC_EEPROM_Initialize() to initialize the EEPROM library first.

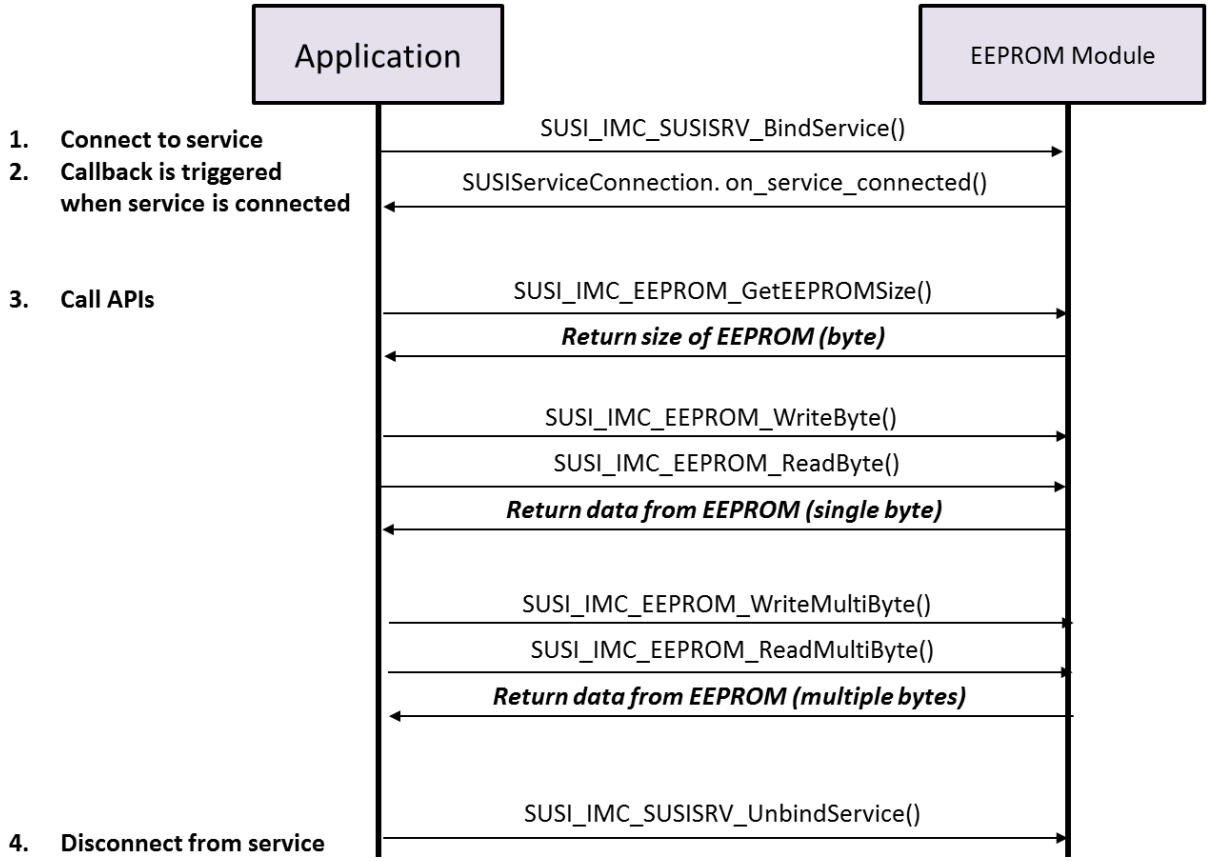
To check EEPROM size, you can use SUSI_IMC_EEPROM_GetEEPROMSize() .

You can read/write single byte or multiple bytes from/to EEPROM through APIs.
For the single byte operation, use SUSI_IMC_EEPROM_ReadByte() to read one byte of data from the target address in EEPROM and use SUSI_IMC_EEPROM_WriteByte() write one byte of data to the target address.
For the multiple bytes operation, you can use SUSI_IMC_EEPROM_ReadMultiByte() to read multi-data from the target address(For Example: Set start address is 00 and address size is 09, then it will get data of

address 00~09) and use SUSI_IMC_EEPROM_WriteMultiByte() to write multi-data to the target address.

To stop the EEPROM module, call SUSI_IMC_EEPROM_Deinitialize() to close API .

5.10.1.2 Android



The class for G sensor APIs for Android is mrm.client.SUSIServiceClient.
For the usage, please also refer to the section SUSI SRV Usage Overview.

After the SUSI service is connected, you can start using the APIs.

To check EEPROM size, you can use SUSI_IMC_EEPROM_GetEEPROMSize() .

You can read/write single byte or multiple bytes from/to EEPROM through APIs.
For the single byte operation, use SUSI_IMC_EEPROM_ReadByte() to read one byte of data from the target address in EEPROM and use SUSI_IMC_EEPROM_WriteByte() write one byte of data to the target address.
For the multiple bytes operation, you can use SUSI_IMC_EEPROM_ReadMultiByte() to read multi-data from the target address(For Example: Set start address is 00 and address size is 09, then it will get data of address 00~09) and use SUSI_IMC_EEPROM_WriteMultiByte() to write multi-data to the target address.

5.10.2 APIs

5.10.2.1 SUSI_IMC_EEPROM_Initialize

Syntax:

Linux / Windows	USHORT **SUSI_IMC_EEPROM_Initialize**();

Description:

Initialize the EEPROM library.

Parameters:

None

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

In the Linux platform, you should ensure the i2c driver(i2c-dev) is loaded before calling this function.

Using the following command to load the i2c driver

```
# modprobe i2c-dev
```

5.10.2.2 SUSI_IMC_EEPROM_Deinitialize

Syntax:

Linux / Windows	USHORT **SUSI_IMC_EEPROM_Deinitialize**(void);

Description:

Deinitialize the EEPROM library.

Parameters:

None

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.10.2.3 SUSI_IMC_EEPROM_GetLibVersion

Syntax:

Windows	USHORT **SUSI_IMC_EEPROM_GetLibVersion**(OUT BYTE version[18]);

Description:

Get current version from the EEPROM library.

Parameters:

version [out]

Point to read buffer that specifies the EEPROM library's current version.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Windows

TREK-572 Windows

TREK-674 Windows

Remark:

The version buffer size must be greater than 18 bytes.

5.10.2.4 SUSI_IMC_EEPROM_GetEEPROMSize

Syntax:

Linux / Windows	USHORT **SUSI_IMC_EEPROM_GetEEPROMSize**(UINT* pROMSize);
Android	short **SUSI_IMC_EEPROM_GetEEPROMSize**(int[] romSize)

Description:

Get the size of EEPROM.

Parameters:

pROMSize [out]

Linux / Windows

Size of EEPROM (byte).

Android

An allocated array of size 1 for storing the returned size of EEPROM(byte).

The return value will be stored at index 0.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.10.2.5 SUSI_IMC_EEPROM_ReadByte

Syntax:

Linux / Windows	USHORT **SUSI_IMC_EEPROM_ReadByte**(BYTE byStartAddr, BYTE* pbyData);
Android	short **SUSI_IMC_EEPROM_ReadByte**(byte startAddr, byte[] data)

Description:

Read one byte of data from the target address in EEPROM.

Parameters:

byStartAddr [in]

Linux / Windows /Android

Specified the 8-bit address, ranging from 0x00 to 0xFF.

pbyData [out]

Linux / Windows

Retrieve data from EEPROM.

Android

An allocated array of size 1 for storing the returned data byte of EEPROM.

The return value will be stored at index 0.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.10.2.6 SUSI_IMC_EEPROM_WriteByte

Syntax:

Linux / Windows	USHORT **SUSI_IMC_EEPROM_WriteByte**(BYTE byStartAddr, BYTE pbyData);
Android	short **SUSI_IMC_EEPROM_WriteByte**(byte startAddr, byte data)

Description:

Write one byte of data to the target address in EEPROM.

Parameters:

byStartAddr [in]

Linux / Windows / Android

Specified the 8-bit address, ranging from 0x00 to 0xFF.

pbyData [in]

Linux / Windows / Android

Data byte to be written to EEPROM.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.10.2.7 SUSI_IMC_EEPROM_ReadMultiByte

Syntax:

Linux / Windows	USHORT **SUSI_IMC_EEPROM_ReadMultiByte**(BYTE byStartAddr, BYTE bySize, BYTE* pbyDataArray, BYTE* pbyRealSize);

Description:

Read multi-byte data from the target address in EEPROM.

Parameters:

byStartAddr [in]

Linux / Windows / Android

Specified the 8-bit address, ranging from 0x00 to 0xFF.

bySize [in]

Linux / Windows / Android

Specified the 8-bit size, ranging from 0x01 to 0x10.

pbyDataArray [out]

Linux / Windows

Pointer to a byte array to be read

Android

An allocated array of desired size for storing the returned data bytes of EEPROM.

The return value will be stored from index 0.

pbyRealSize [out]

Linux / Windows

The number of bytes actual read.

Android

An allocated array of size 1 for storing the number of bytes actual read.

The return value will be stored from index 0.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.10.2.8 SUSI_IMC_EEPROM_WriteMultiByte

Syntax:

Linux / Windows	USHORT **SUSI_IMC_EEPROM_WriteMultiByte** (BYTE byStartAddr, BYTE bySize, BYTE* pbyDataArray, BYTE* pbyRealSize);
Android	short **SUSI_IMC_EEPROM_WriteMultiByte**(byte startAddr, int size, byte[] dataArray, int[] realSize)

Description:

Write multi-byte data to the target address in EEPROM.

Parameters:

byStartAddr [in]

Linux / Windows / Android

Specified the 8-bit address, ranging from 0x00 to 0xFF.

bySize [in]

Linux / Windows / Android

Specified the 8-bit size, ranging from 0x01 to 0x10.

pbyDataArray [in]

Linux / Windows

Pointer to a byte array to be written.

Android

An allocated array of desired size which store the data bytes to be written.

pbyRealSize [out]

Linux / Windows

The number of bytes actual written.

Android

An allocated array of size 1 for storing the number of bytes actual written.

The return value will be stored from index 0.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

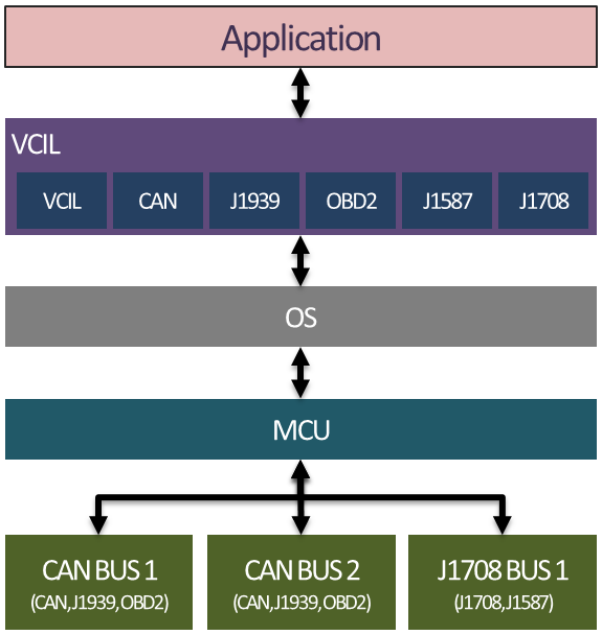
TREK-674 Linux / Windows

Remark:

None

5.11 VCIL

5.11.1 Usage Overview

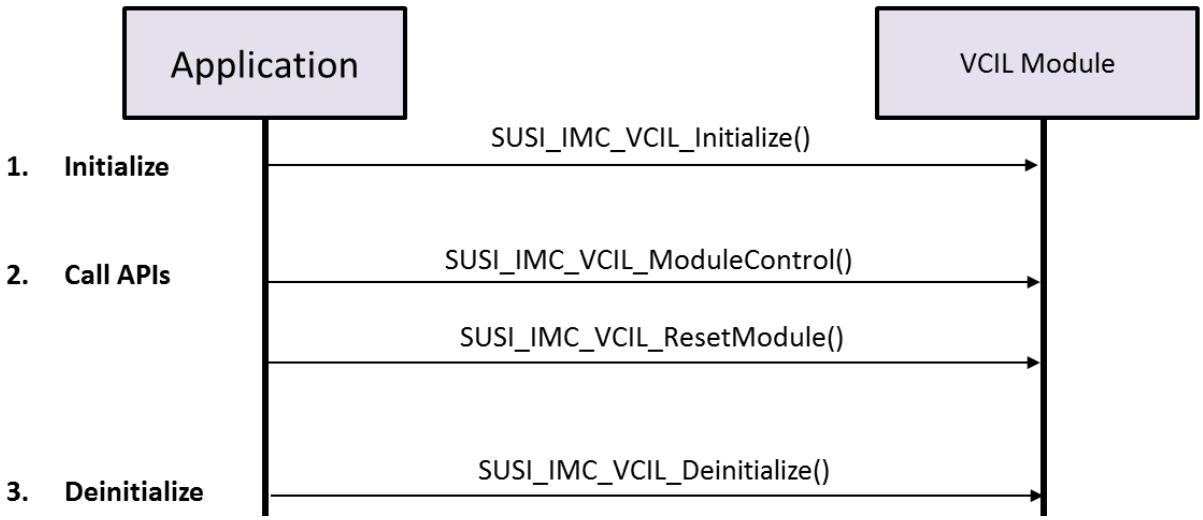


TREK devices are equipped with a VCIM MCU which controls the data flow of vehicle communication protocols (CAN, J1939, OBD2, J1708, J1939) on the data buses.

To make application able to access the MCU and use vehicle protocols easier, TREK SDK introduces VCIL(Vehicle Communication Interface Layer) API modules which provide interfaces for controlling each protocols to achieve various purposes[*1].

The following section describes the usage of VCIL APIs.

5.11.1.1 Windows/ Linux/ Android



To start using VCIL APII module and related modules, you should first first call SUSI_IMC_VCIL_Initialize() /SUSI_IMC_VCIL_InitializeEx() before using the other VCIL API. To stop using VCIL module, you must call SUSI_IMC_VCIL_Deinitialize() to close API.

To set the activated protocols for each channel, you can use SUSI_IMC_VCIL_ModuleControl() to set.

The available modes for each CAN Channel are as followings:

- (0x00) **VCIL_MODE_CAN -** CAN protocol without filter **(DEFAULT)**
- (0x01) **VCIL_MODE_CAN_WITH_MASK -** CAN protocol with filter
- (0x02) **VCIL_MODE_J1939 -** J1939 protocol without filter
- (0x03) **VCIL_MODE_J1939_WITH_FILTER -** J1939 protocol with filter
- (0x04) **VCIL_MODE_ODB2 -** OBD2 protocol without filter
- (0x05) **VCIL_MODE_ODB2_WITH_FILTER** - OBD2 protocol with filter

The available modes for each J1708 Channel are as followings:

- (0x00) **VCIL_MODE_J1708** - J1708 protocol with filter **(DEFAULT)**
- (0x01) **VCIL_MODE_J1587** - J1587 protocol with filter

For example, to activate CAN with filter on CAN channel 1, J1939 without filter on CAN channel 2 and J1708 without filter on J1708 channel 1, you can call the API with following parameters - SUSI_IMC_VCIL_ModuleControl(VCIL_MODE_CAN_WITH_MASK, VCIL_MODE_J1939, VCIL_MODE_J1708).

To reset the MCU, you can use SUSI_IMC_VCIL_ResetModule().

5.11.2 Enumeration

5.11.2.1 Windows/Linux

VCIL_CAN_CHANNEL_MODE enum

- (0x00) **VCIL_MODE_CAN** - Active at RAW CAN mode. **(DEFAULT)**
- (0x01) **VCIL_MODE_CAN_WITH_MASK** - Active at RAW CAN mode with mask.
- (0x02) **VCIL_MODE_J1939** - Active at J1939 mode.
- (0x03) **VCIL_MODE_J1939_WITH_FILTER** - Active at RAW CAN mode with filter.
- (0x04) **VCIL_MODE_ODB2** - Active at OBD2 mode.
- (0x05) **VCIL_MODE_ODB2_WITH_FILTER** - Active at RAW CAN mode with filter.

VCIL_J1708_CHANNEL_MODE enum

- (0x00) **VCIL_MODE_J1708** - Active at J1708 mode. **(DEFAULT)**
- (0x01) **VCIL_MODE_J1587** - Active at J1587 mode.

5.11.2.2 Android

VCIL_CAN_CHANNEL_MODE enum

class:

mrm.define.IMC_ENUM.**VCIL_CAN_CHANNEL_MODE**

Enum:

Name	Type	value	Comment
VCIL_MODE_CAN	byte	0x00	CAN protocol.
VCIL_MODE_CAN_WITH_MASK	byte	0x01	CAN protocol with ID filter
VCIL_MODE_J1939	byte	0x02	J1939 protocol
VCIL_MODE_J1939_WITH_FILTER	byte	0x03	J1939 protocol with PGN filter
VCIL_MODE_OBD2	byte	0x04	OBD2 protocol
VCIL_MODE_OBD2_WITH_FILTER	byte	0x05	OBD2 protocol with PID filter

Remark:

Use the method **getValue()** to get the enum value.

ex: MRM_ENUM.VCIL_CAN_CHANNEL_MODE.VCIL_MODE_CAN_WITH_MASK**.getValue()** returns 1.

Please refer to sample code for detailed usage.

VCIL_J1708_CHANNEL_MODE enum

class:

mrm.define.IMC_ENUM.**VCIL_J1708_CHANNEL_MODE**

Enum:

Name	Type	value	Comment
VCIL_MODE_J1708	byte	0x00	J1708
VCIL_MODE_J1587	byte	0x01	J1587

Remark:

Use the method **getValue()** to get the enum value.

ex: MRM_ENUM.VCIL_J1708_CHANNEL_MODE.VCIL_MODE_J1587**.getValue()** returns 1.

Please refer to sample code for detailed usage.

5.11.3 Constant

5.11.3.1 Android

Class:
mrm.define.**IMC_CONSTANT**

Fields:

Field Name	Type	Value
VCIL_MAXIMUM_FIRMWARE_VERSION_LENGTH	int	2
MAX_CAN_MESSAGE_BUFFER_SIZE	int	8
MAX_J1939_MESSAGE_BUFFER_SIZE	int	64
MAX_J1939_FILTER_NUMBER	int	128
J1939_TRANSMIT_CONFIG_NAME_LEN	int	8
MAX_OBD2_MESSAGE_BUFFER_SIZE	int	64
MAX_OBD2_FILTER_NUMBER	int	256
MAX_J1708_MESSAGE_BUFFER_SIZE	int	20
MAX_J1708_FILTER_NUMBER	int	256
MAX_J1587_MESSAGE_BUFFER_SIZE	int	20
MAX_J1587_FILTER_NUMBER	int	512

5.11.4 APIs

5.11.4.1 SUSI_IMC_VCIL_Initialize

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VCIL_Initialize** (char* port)
Android	short **SUSI_IMC_VCIL_Initialize** (String port)

Description:

Initialize the VCIL library.

Parameters:

port [in]

Please input the port number for windows system.

Example, if your port is COM7, you would input the “7”.

Please input the path of device of mudule for Linux platform.

Example, you would input the “/dev/ttyO0” on TREK-520.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

It shown the UART mapping table as following:

Hardware Platform	OS Platform	Port ID/PATH
TREK-520	WinCE	*port = 8
TREK-520	Linux	port = “/dev/ttyO0”
TREK-674/570/572	Windows	*port = 7
TREK-674/570/572	Linux	port =”/dev/ttyA0”
TREK-570	Android	port =”/dev/ttyA0”

5.11.4.2 SUSI_IMC_VCIL_InitializeEx

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VCIL_InitializeEx** (char* can_port, int baudrate)
Android	short **SUSI_IMC_VCIL_InitializeEx** (String port, int baudrate)

Description:

Initialize the VCIL library by addition argument.

Parameters:

port [in]

Please input the port number for windows system.

Example, if your port is COM7, you would input the “7”.

Please input the path of device of module for Linux platform.

Example, you would input the “/dev/ttyO0” on TREK-520.

baudrate [in]

communication to firmware baud rate with 921600 or 460800

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

It shown the UART mapping table as following:

Hardware Platform	OS Platform	Port ID/PATH
TREK-520	WinCE	*port = 8
TREK-520	Linux	port = “/dev/ttyO0”
TREK-674/570/572	Windows	*port = 7
TREK-674/570/572	Linux	port =”/dev/ttyA0”
TREK-570	Android	port =”/dev/ttyA0”

5.11.4.3 SUSI_IMC_VCIL_Deinitialize

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VCIL_Deinitialize** (void);
Android	short **SUSI_IMC_VCIL_Deinitialize** ()

Description:

Deinitialize the VCIL library.

Parameters:

None

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

Notify: Call this function before unloading the VCIL library from memory.

5.11.4.4 SUSI_IMC_VCIL_GetLibVersion

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VCIL_GetLibVersion** (OUT BYTE version[18]);

Description:

Get current version from the VCIL library.

Parameters:

version [out]

Point to read buffer that specifies the VCIL library's current version.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

The version buffer size must be greater than 18 bytes.

5.11.4.5 SUSI_IMC_VCIL_GetFWVersion

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VCIL_GetFWVersion** (OUT PBYTE version);
Android	short **SUSI_IMC_VCIL_GetFWVersion** (byte[] version)

Description:

Get the firmware information from VCIL.

Parameters:

pversion [out]

Linux / Windows

Points to a struct that specifies the firmware information.

Android:

An allocated byte array of size VCIL_MAXIMUM_FIRMWARE_VERSION_LENGTH (2) large for storing the firmware version code.

You can get the firmware major version decimal at index 0 and minor version decimal at index 1.

Please refer to the sample code for detailed usage.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

The buffer size must more then the 2 bytes. If you input the buffer size less then 2 bytes, this API will have abort or exception.

5.11.4.6 SUSI_IMC_VCIL_ResetModule

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VCIL_ResetModule** (void);
Android	short **SUSI_IMC_VCIL_ResetModule**()

Description:

Reset the VCIL module.

Parameters:

None

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

After call this function, the CPU will reset and all the internal setting return to default.

Note: All filter will be clear.

5.11.4.7 SUSI_IMC_VCIL_ModuleControl

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_VCIL_ModuleControl** (IN VCIL_CAN_CHANNEL_MODE can_channel_1, VCIL_CAN_CHANNEL_MODE can_channel_2, VCIL_J1708_CHANNEL_MODE j1708_channel);
Android	short **SUSI_IMC_VCIL_ModuleControl**(int can_channel_1, int can_channel_2, int j1708_channel)

Description:

Set a configuration of the VCIL physical channel to specify protocol.

Parameters:

can_channel_1 [in]

Linux / Windows

Protocol of CAN channel 1.

Android:

Protocol of CAN channel 1.

Available values are enumerated in class VCIL_CAN_CHANNEL_MODE

can_channel_2 [in]

Linux / Windows

Protocol of CAN channel 2.

Android:

Protocol of CAN channel 2.

Available values are enumerated in class VCIL_CAN_CHANNEL_MODE

j1708_channel [in]

Linux / Windows

Protocol of J1708 channel 1.

Android:

Protocol of J1708 channel 1.

Available values are enumerated in class VCIL_J1708_CHANNEL_MODE

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

The filter setting will not be kept on mode changed. Application should set the filter again on mode changed.

5.12 CAN

5.12.1 Usage Overview

TREK device supports data transmission of CAN protocol and provides filter functions.
The SDK hold a internal buffer for received CAN messages with limited size. You can use the read API to consume the buffer and get received message. When the internal buffer become full before application consume it, the oldest message will be removed and new message will be added.

The following section describe the usage of corresponding APIs.

5.12.1.1 Basic Usage

This section describes the basic usage of CAN API module.

Windows/Linux

To start using CAN APIs you should first initialize the APIs and activate CAN protocol on specific CAN channel.

Please refer to the VCIL usage for how to initialize and configure CAN channels.

Android

The class for CAN APIs for Android is **mrm.VCIL**
To start using CAN APIs you should first initialize the APIs and activate CAN protocol on specific CAN channel .

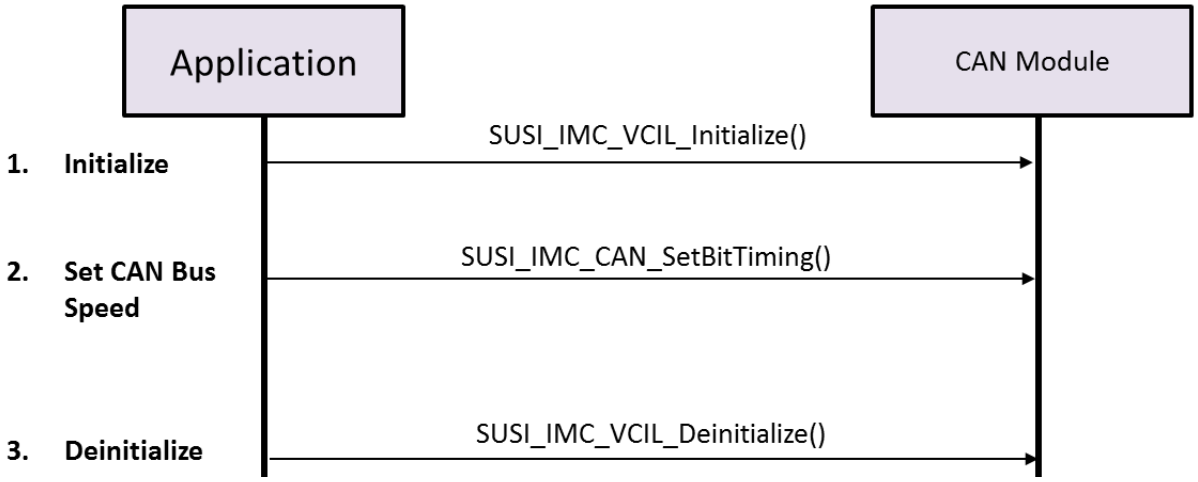
Please refer to the VCIL Usage for basic usage and for how to initialize and configure CAN channels.

5.12.1.2 CAN Bus Speed Setting

To start data transmission of CAN, J1939, OBD2 protocol on CAN bus, you must first configure the CAN bus speed for MCU.

This section describes how to set CAN bus speed through SDK.

Windows/ Linux/ Android



The way to configure CAN bus speed is simple, You just need to call SUSI_IMC_CAN_SetBitTiming() after API is initialized. The you can start data transmission.

5.12.1.3 CAN Message Reading

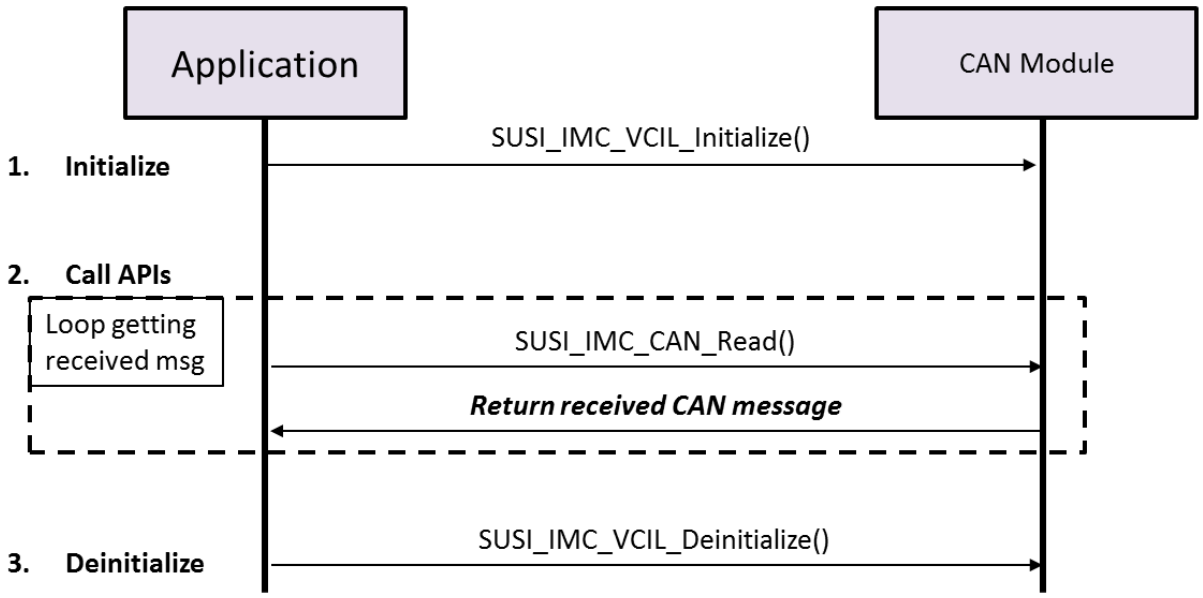
This section describe how to read CAN messages from CAN bus by using SDK APIs.

The CAN bus speed must be set correctly before you do read/write operation. Please refer to CAN Bus Speed Setting section for the details.

Windows/Linux

You can implement your CAN message reading application in either **Polling** or **Event Handling** style.

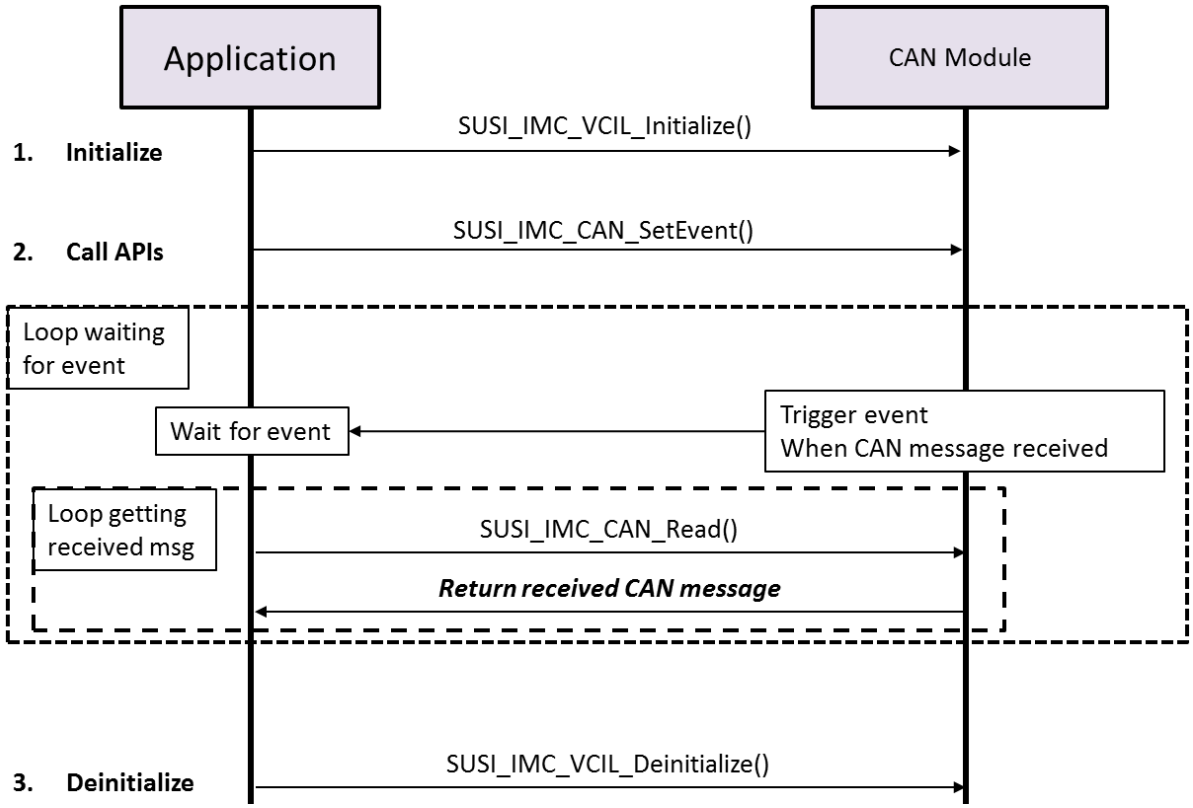
- **Polling:**



The most simple way to get received message is to hold a loop in your APP and keep calling SUSI_IMC_CAN_Read() to get received message from SDK internal buffer.

The advantage of reading messages in polling style is that it is simple and relatively lower overhead (i.e. no event handling) to read a message.
The disadvantage is that you need to keep your APP's process reading message even when there is actually no message in SDK internal buffer, which may result in unnecessary power consuming.

- **Event Handling:**



To read messages in event handling style, you need to call SUSI_IMC_CAN_SetEvent() first to register an event to CAN API module. When MCU receive a CAN message from CAN bus, SDK adds the received message to internal buffer and trigger the registered event.

You need to hold an event listening loop in your APP to handle the registered event. When the event is triggered, you can then run a loop to calling **SUSI_IMC_CAN_Read()** to get and consume the received messages from the internal buffer. After all messages in buffer are consumed, you should keep listening the registered event. Please refer to the sample code for the details of implementation.

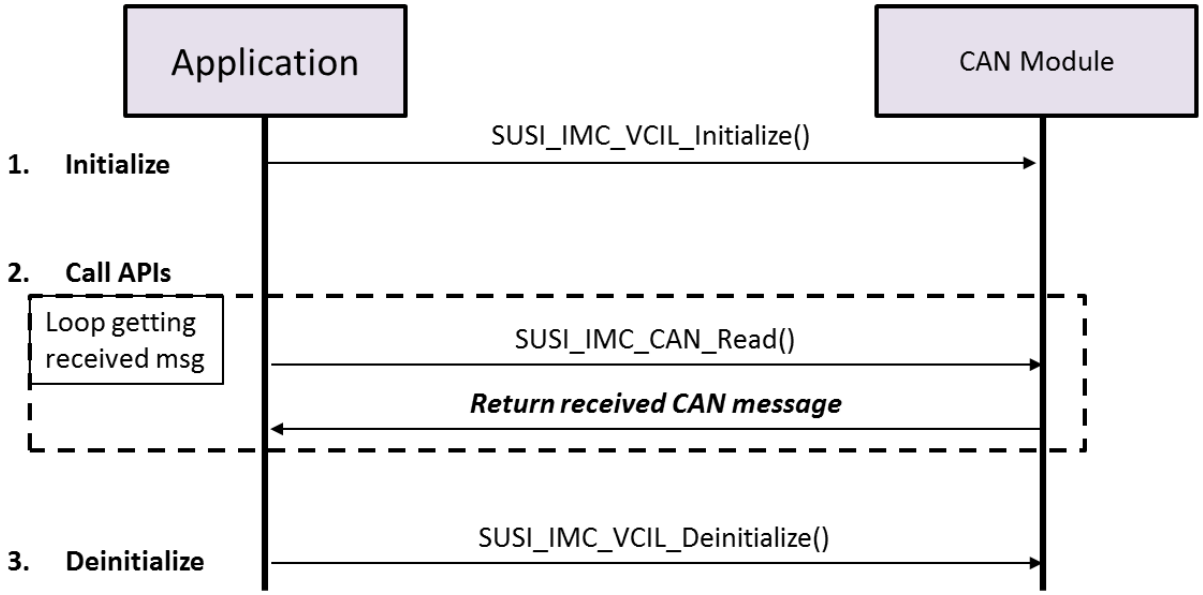
The advantage of reading messages in event handling style is that APP's process only work when there are messages can be read, which relatively cost less system resources and power.
The disadvantage is that there would be some overhead of event handling to read a message.

Please refer to the sample code for implementation details.

Android

You can implement your CAN message reading application in either **Polling** or **Event Handling** style.

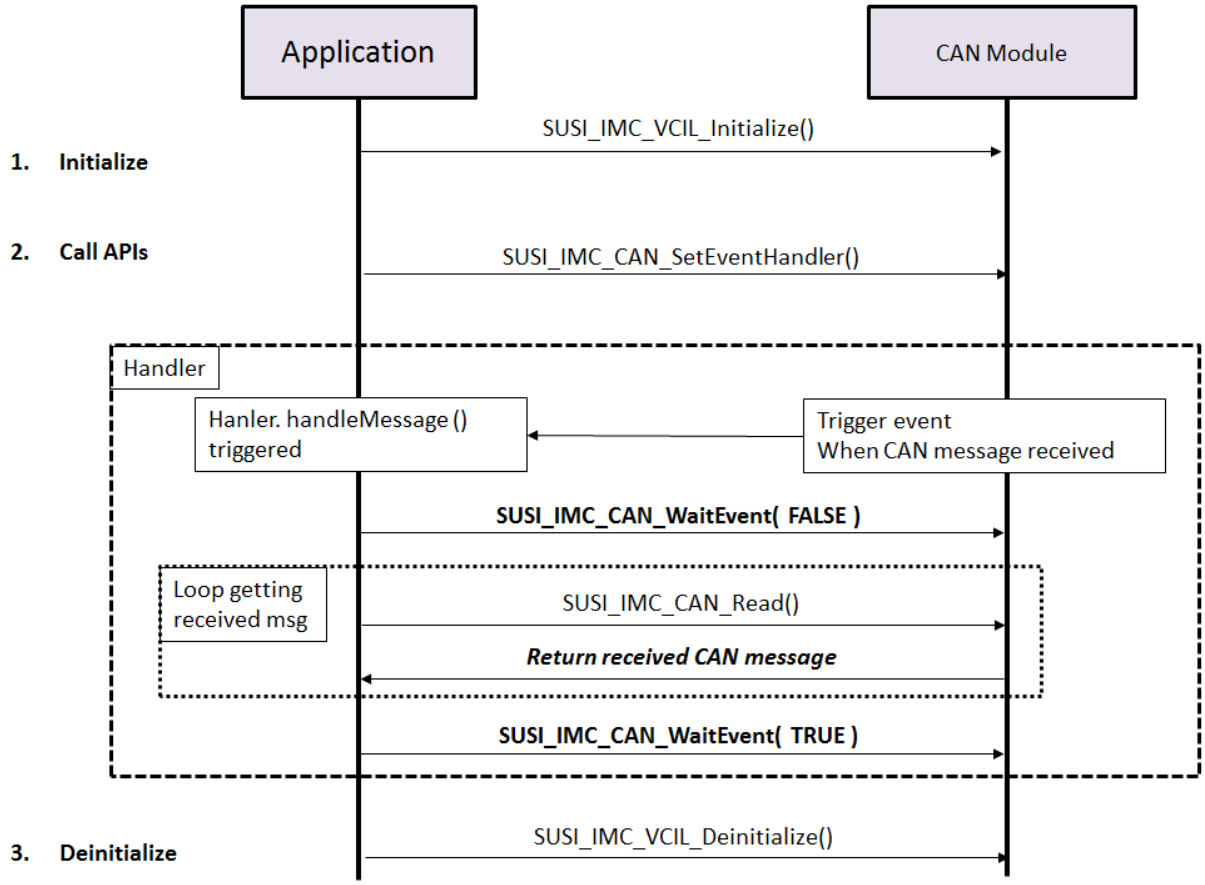
- **Polling:**



The most simple way to get received message is to hold a loop in your application and keep calling SUSI_IMC_CAN_Read()) to get received message(s) from SDK internal buffer.

The advantage of reading messages in polling style is that it is simple and relatively lower overhead (i.e. no event handling) to read message(s).
The disadvantage is that you need to keep your application process reading message even when there is actually no message in SDK internal buffer, which may result in unnecessary power consuming.

- **Event Handling:**



TREK SDK leverage the Android Handler mechanism to inform CAN message receive event.

To read messages in event handling style, you need to create an instance of Handler and call SUSI_IMC_CAN_SetEventHandler() first to register the Handler instance to VCIL. When MCU receive a CAN message from CAN bus, VCIL adds the received message to internal buffer and trigger the event to inform registered handler.

When the event is triggered, the handleMessage() callback of registered handler instance will be triggered. you can then run a loop to calling SUSI_IMC_CAN_Read() to get and consume the received messages from the internal buffer.

Due to that VCIL service will trigger event whenever it receive a CAN message, to avoid from getting multiple unnecessary events, you should call SUSI_IMC_CAN_WaitEvent(FALSE) to ask VCIL service temporarily stop pass event to handler before your APP start the alarm data getting loop. After all messages in buffer are consumed, you should call SUSI_IMC_CAN_WaitEvent(TRUE) to ask VCIL service continue the event informing.

If you do not need to listen to alarm event anymore, you should call SUSI_IMC_CAN_UnsetEventHandler() to unregister the Handler instance from VCIL service.

The advantage of reading messages in event handling style is that application process only work when

there are messages can be read, which relatively cost less system resources and power.

The disadvantage is that there would be some overhead of event handling to read a message.

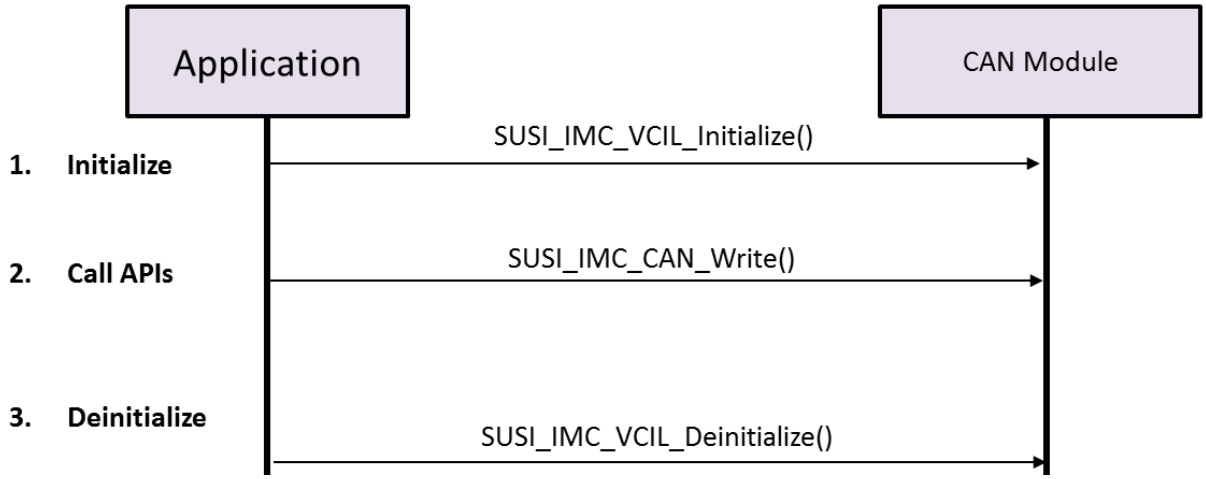
Please refer to the sample code for implementation details.

5.12.1.4 CAN Message Writing

This section describe how to write CAN messages to CAN bus by using SDK APIs.

The CAN bus speed must be set correctly before you do read/write operation. Please refer to CAN Bus Speed Setting section for the details.

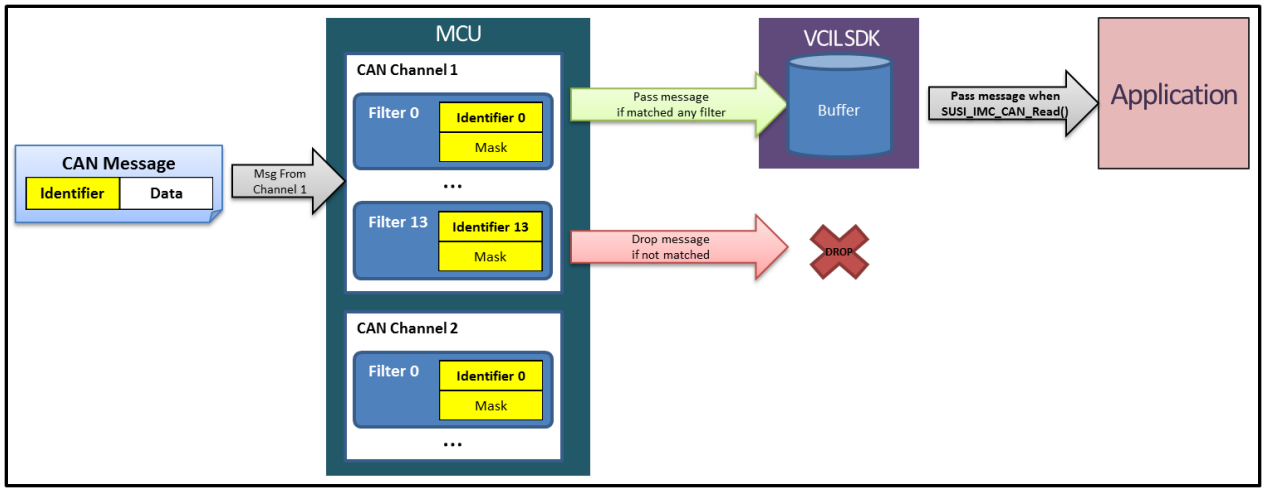
Windows/ Linux/ Android



To write CAN message to CAN bus, you can call SUSI_IMC_CAN_Write() with a IMC_CAN_MSG_OBJECT structure which contains the data you want to write.

5.12.1.5 CAN Acceptance filter Settings

In the car system there might be many nodes on the bus, the number of CAN messages transmitted to the bus might be enormous. Due to the performance issues and application purpose concerns, it might not be a good idea to try to process all messages on the bus. To focus on the messages you interest in, you can use the filter functions provided by VCIL MCU.



The VCIL MCU provides 14 configurable CAN identifier filter banks for filtering the incoming messages for each CAN channel. The filters act as "white list". If the CAN channel is configured to work with filters, the MCU which will only receive the CAN messages which match the filter conditions and drop others.

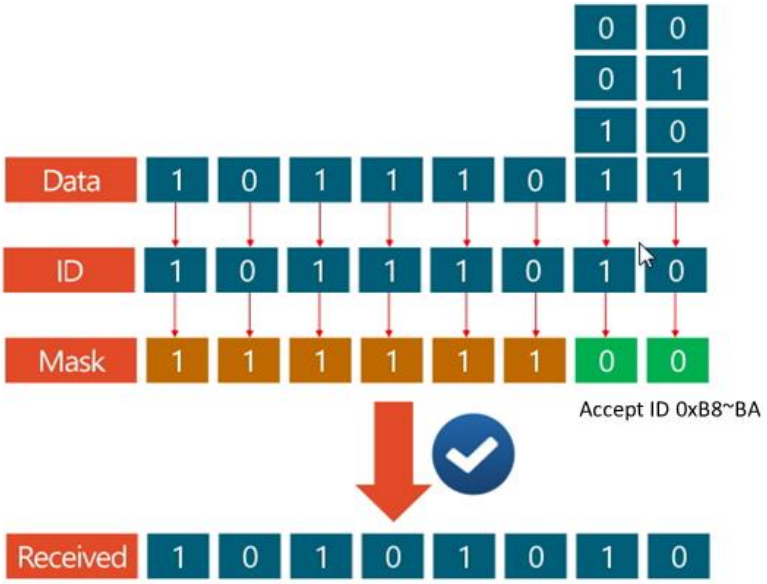
To enable the filter function, you **must configure the CAN physical channel to VCIL_MODE_CAN_WITH_MASK, to work with filter first**. Please refer to the Basic Usage of VCIL section.

For each filter, a identifier value and mask is set. The identifier value defines a desired identifier and the mask defined the bits of identifier the filter care about. If the mask of filter is set to 0, then it means the filter "don't care" any bit of the identifier and all CAN messages from the bus will be passed through this filter. When VCIL MCU get a CAN message from bus, it AND the identifier of message and the identifier of each filter with mask to test whether the message should be passed or not.

The following are examples of filter mask setting.

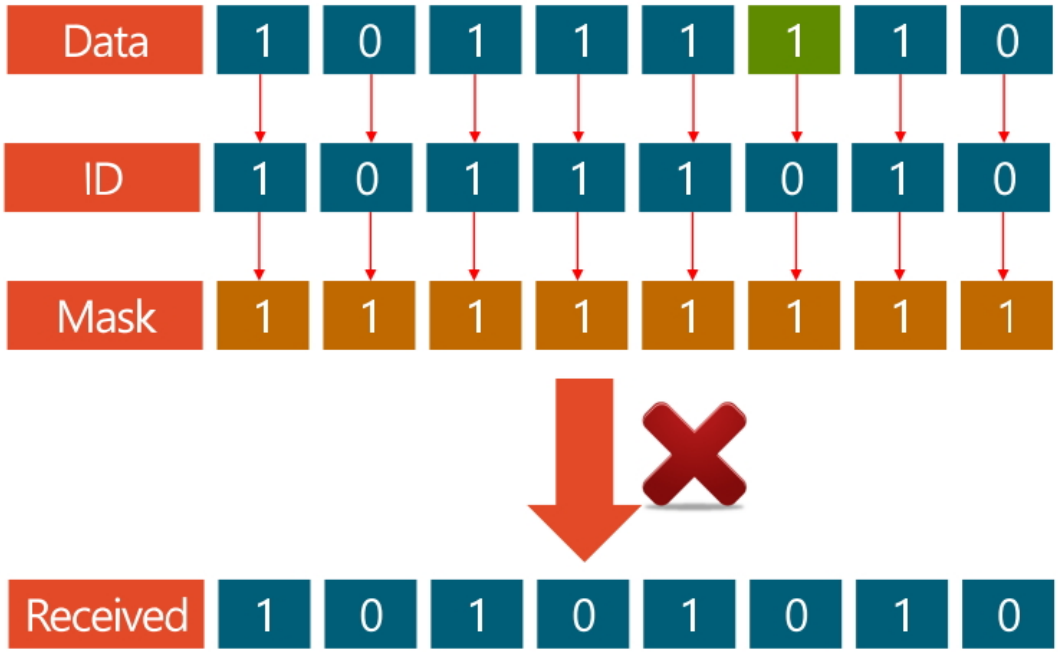
- Accepted example

If we set a filter with mask value = [1 1 1 1 1 1 0 0] (0xFC) and identifier = [1 0 1 1 1 0 1 0](0xBA), then MCU will care the first 6 bits of the identifier.
Thus MCU will only accept messages with identifier 0xB8~BA. (Binary:1011 = Hex:B)



- Unaccepted example

If we set a filter with mask value = [1 1 1 1 1 1 1 1] (0xFF) and identifier = [1 0 1 1 1 0 1 0](0xBA), then MCU will care the all bits of the identifier.
Thus MCU will only accept messages with identifier 0xBA.



Once APP sets CAN channel to “CAN with mask” mode, **the VCIL MCU will automatically add a filter with mask = 0 to filter bank 0. So, all messages will be passed by default.** Once application add a filter rule to bank 0, the default all-pass filter is overwritten and only message with specified identified will be passed.

The following shows examples of filtering.

- **Example - Single filter**

CAN MASK Configuration	Receive CAN data	Result
Filter 1:	10111010 (0xBA)	**Accept**

CAN ID=10111010b (0xBA) Mask = 11111111b (0xFF)		
Filter 1: CAN ID=10111010b (0xBA) Mask = 11111111b (0xFF)	10111011 (0xBB)	**Drop**

CAN MASK Configuration	Receive CAN data	Result
Filter 1: CAN ID=10111010b (0xBA) Mask = 11111100b (0xFC)	10111000 (0xB8)	**Accept**
Filter 1: CAN ID=10111010b (0xBA) Mask = 11111100b (0xFC)	10111001 (0xB9)	**Accept**
Filter 1: CAN ID=10111010b (0xBA) Mask = 11111100b (0xFC)	10111010 (0xBA)	**Accept**
Filter 1: CAN ID=10111010b (0xBA) Mask = 11111100b (0xFC)	10111100 (0xBC)	**Drop**

- **Example - Multiple filters**

You can set multiple filter at same time for multiple range of targets. For example, ID 0x28~0x37 and 0x3000~0x31FF and 0x1600

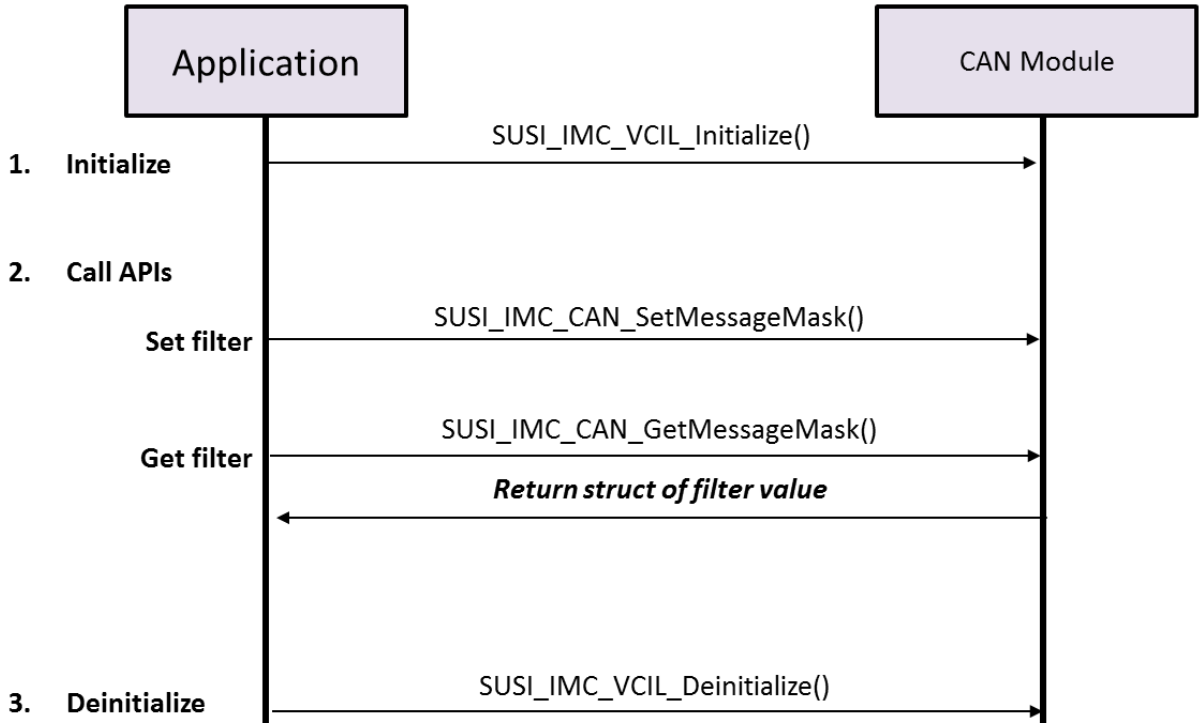
CAN MASK Configuration	Receive CAN data	Result
Filter 1: CAN ID=0000000000101000b (0x0028), Mask = 1111111111111110b (0xFFFE) Filter 2: CAN ID=0000000000110000b (0x0030) Mask = 1111111111111000b (0xFFF8) Filter 3: CAN ID=0011000000000000b (0x3000) Mask = 1111111000000000b (0xFE00) Filter 4: CAN ID=0001011000000000b (0x1600) Mask = 1111111111111111b (0xFFFF)	00101000b(0x29)	**Accept**
Filter 1: CAN ID=0000000000101000b (0x0028) Mask = 1111111111111110b (0xFFFE) Filter 2: CAN ID=0000000000110000b (0x0030) Mask = 1111111111111000b (0xFFF8) Filter 3: CAN ID=0011000000000000b (0x3000) Mask = 1111111000000000b (0xFE00) Filter 4: CAN ID=0001011000000000b (0x1600) Mask = 1111111111111111b (0xFFFF)	01001000b(0x48)	**Drop**
Filter 1: CAN ID=0000000000101000b (0x0028) Mask = 1111111111111110b (0xFFFE) Filter 2: CAN ID=0000000000110000b (0x0030) Mask = 1111111111111000b (0xFFF8)	00110001000000010b(0x3102)	**Accept**

Filter 3: CAN ID=0011000000000000b (0x3000) Mask = 1111111000000000b (0xFE00) Filter 4: CAN ID=0001011000000000b (0x1600) Mask = 1111111111111111b (0xFFFF)		
Filter 1: CAN ID=0000000000101000b (0x0028) Mask = 1111111111111110b (0xFFFE) Filter 2: CAN ID=0000000000110000b (0x0030) Mask = 1111111111111000b (0xFFF8) Filter 3: CAN ID=0011000000000000b (0x3000) Mask = 1111111000000000b (0xFE00) Filter 4: CAN ID=0001011000000000b (0x1600) Mask = 1111111111111111b (0xFFFF)	0011001000000000b(0x3200)	**Drop**

Windows/ Linux/ Android

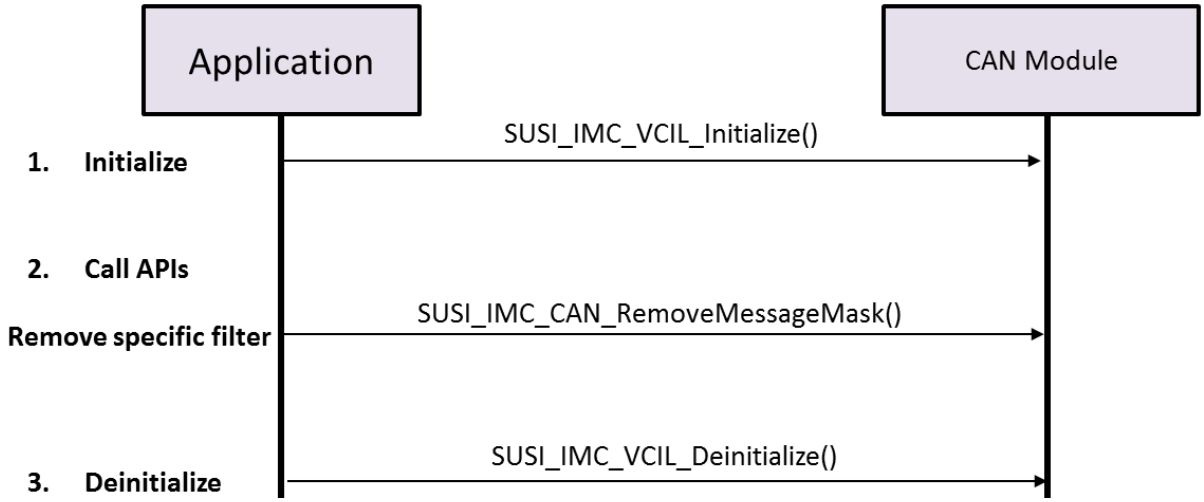
- **To get/set filter:**

Call SUSI_IMC_CAN_SetMessageMask() to set the mask and call SUSI_IMC_CAN_GetMessageMask() to get mask.



- **To remove specific filter:**

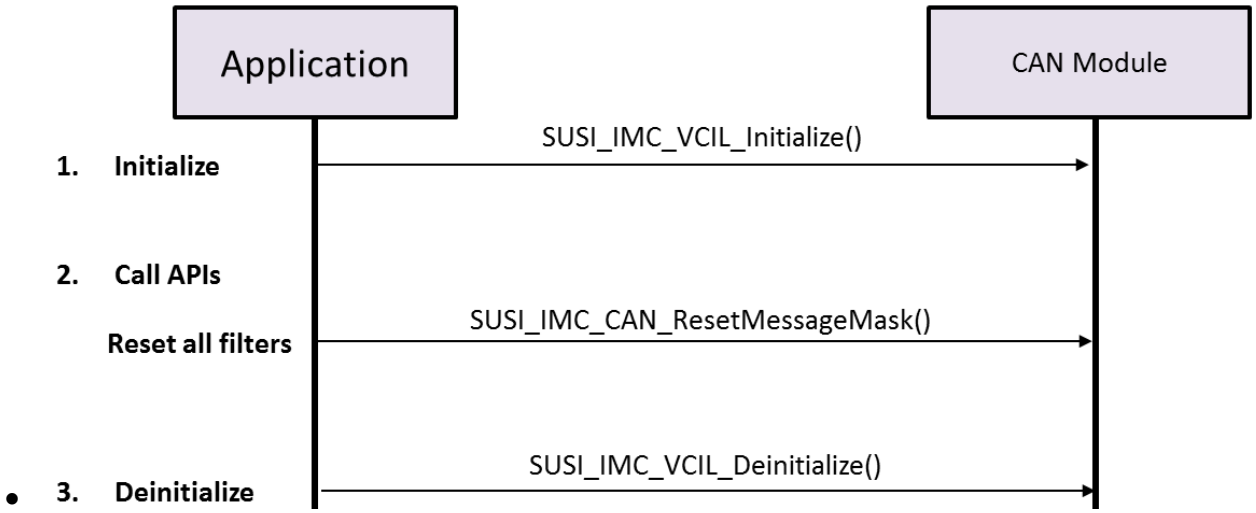
Call SUSI_IMC_CAN_RemoveMessageMask() to remove filter of specified filter ID.



- **To remove all filter:**

Call SUSI_IMC_CAN_ResetMessageMask() to reset all filter.

The default mask add to bank 0 with mask equal 0.


Application
CAN Module
1. Initialize
SUSI_IMC_VCIL_Initialize()
2. Call APIs
Reset all filters
SUSI_IMC_CAN_ResetMessageMask()
3. Deinitialize
SUSI_IMC_VCIL_Deinitialize()

5.12.2 Enumeration

5.12.2.1 Windows/Linux

CAN_MESSAGE_TYPE enum

- (0x01) **CAN_MESSAGE_STANDARD** - CAN 2.0a (Standard message)
- (0x02) **CAN_MESSAGE_EXTENDED** - CAN 2.0b (Extend message)
- (0x04) **CAN_MESSAGE_RTR** - Request Remote transmission frame
- (0x08) **CAN_MESSAGE_STATUS** - CAN message status.

CAN_SPEED enum

- (0) **CAN_SPEED_125K** - 125k bit/s
- (1) **CAN_SPEED_250K** - 250k bit/s
- (2) **CAN_SPEED_500K** - 500k bit/s
- (3) **CAN_SPEED_1M** - 1M bit/s
- (4) **CAN_SPEED_200K** - 200k bit/s
- (5) **CAN_SPEED_100K** - 100k bit/s
- (0xFE) **CAN_SPEED_INIT** - initiation (bus-off)
- (0xFF) **CAN_SPEED_USER_DEFINE** - User define

CAN_BUS_MODE enum

- (0x00) **CAN_BUS_MODE_NORMAL** -
  CAN controller operates in normal mode. In normal mode, CAN controller synchronize the bit traffic on the CAN bus and is able to receive/transmit CAN messages.

- (0x01) **CAN_BUS_MODE_SILENCE** -
  CAN controller operates in silence mode. In silence mode, the CAN controller is only able to receive valid data frames and remote request frames and NOT able to transmit. The CAN controller only monitor the bit traffic on bus without interfering the bus (e.g. keep in recessive state) and NO dominant bit will be sent (i.e. Acknowledge Bits, Error Frames) .
  Silence mode can be used to monitor the traffic on a CAN bus without interfering the bus.

- (0x02) **CAN_BUS_MODE_INIT** -
  CAN controller operates in initiation mode. In Initialization Mode, the CAN controller stops transmitting and receiving to/from the CAN bus, and dose not interfere the bus traffic (keeps bus output in recessive state).

5.12.2.2 Android

CAN_MESSAGE_TYPE enum

class:

mrm.define.IMC_ENUM.**CAN_MESSAGE_TYPE**

Enum:

Name	Type	value	Comment
CAN_MESSAGE_STANDARD	byte	0x01	CAN 2.0a (Standard message)
CAN_MESSAGE_EXTENDED	byte	0x02	CAN 2.0b (Extend message)
CAN_MESSAGE_RTR	byte	0x04	Request Remote transmission frame
CAN_MESSAGE_STATUS	byte	0x08	CAN message status.

Remark:

Use the method **getValue()** to get the enum value.

ex: MRM_ENUM.CAN_MESSAGE_TYPE.CAN_MESSAGE_STANDARD**.getValue()** returns 1.

Please refer to sample code for detailed usage.

CAN_SPEED enum

class:

mrm.define.IMC_ENUM.**CAN_SPEED**

Enum:

Name	Type	value	Comment
CAN_SPEED_125K	int	0	125k bit/s
CAN_SPEED_250K	int	1	250k bit/s
CAN_SPEED_500K	int	2	500k bit/s
CAN_SPEED_1M	int	3	1M bit/s
CAN_SPEED_200K	int	4	200k bit/s
CAN_SPEED_100K	int	5	100k bit/s

Remark:

Use the method **getValue()** to get the enum value.

ex: MRM_ENUM.CAN_SPEED.CAN_SPEED_250K**.getValue()** returns 1.

Please refer to sample code for detailed usage.

CAN_BUS_MODE enum

class:

mrm.define.IMC_ENUM.**CAN_BUS_MODE**

Enum:

Name	Type	value	Comment
CAN_BUS_MODE_NORMAL	int	0	CAN controller operates in normal mode. In normal mode, CAN controller synchronize the bit traffic on the CAN bus and is able to receive/transmit CAN messages.
CAN_BUS_MODE_SILENCE	int	1	CAN controller operates in silence mode. In silence mode, the CAN controller is only able to receive valid data frames and remote request frames and NOT able to transmit. The CAN controller only monitor the bit traffic on bus without interfering the bus (e.g. keep in recessive state) and NO dominant bit will be sent (i.e. Acknowledge Bits, Error Frames) . Silence mode can be used to monitor the traffic on a CAN bus without interfering the bus.
CAN_BUS_MODE_INIT	int	2	CAN controller operates in initiation mode. In Initialization Mode, the CAN controller stops transmitting and receiving to/from the CAN bus, and keep the bus output in recessive status.

Remark:

Use the method **getValue()** to get the enum value.

ex: MRM_ENUM.CAN_BUS_MODE.CAN_BUS_MODE_SILENCE**.getValue()** returns 1.

Please refer to sample code for detailed usage.

5.12.3 Structure/Class

5.12.3.1 Windows/Linux

IMC_CAN_MSG_OBJECT Structure

Syntax:

```
typedef struct
{
        int can_bus_number
        ULONG id;
        CAN_MESSAGE_TYPE message_type;
        BYTE buf[8]
        BYTE buf_len;
}IMC_CAN_MSG_OBJECT, *PIMC_CAN_MSG_OBJECT;
```

Description:

This structure defines the CAN messages.

Members:

can_bus_number

The range is 1 to 2

Id

Identifier of the CAN message. Range from 0 to 7FFh for 11 bit identifiers and from 0 to 1FFFFFFFh for 29 bit identifiers (CAN specification 2.0b)

message_type

CAN_MESSAGE_TYPE enum

buf

Point to a receive buffer.

Buf_len

Data length code of the message (range from 0 to 8).

IMC_CAN_MASK_OBJECT Structure

Syntax:

```
typedef struct
{
        int can_bus_number;
        int mask_number;
        CAN_MESSAGE_TYPE message_type;
        ULONG id;
        ULONG mask;
}IMC_CAN_MASK_OBJECT, *PIMC_CAN_MASK_OBJECT;
```

Description:

This data structure defines the hardware CAN mask.

Members:

can_bus_number

CAN bus number

mask_number

Mask number. There are a total of 14 units of the mask.

Range: 0~13.

message_type

CAN_MESSAGE_TYPE enum

Id

Identifier of the CAN message. Range from 0 to 7FFh for 2.0A and from 0 to 1FFFFFFFh for 2.0B.

mask

Message filter mask. BitX = 0 don't care, BitX = 1 care

IMC_CAN_ERROR_STATUS_OBJECT Structure

Syntax:

```
typedef struct
{
    int rec; // receive error counter
    int tec; // transmit error counter
    int last_error_code; // last error code
    int error_flag;
} IMC_CAN_ERROR_STATUS_OBJECT, *PIMC_CAN_ERROR_STATUS_OBJECT;
```

Description:

This data structure defines the CAN bus error status.

Members:

rec

Receive error counter. When the CAN controller receive the bus error, the counter will increase.

tec

Transmit error counter. When transmit CAN message encounter error, the counter will increase.

last_error_code

Define the last error code on bus

0: No Error

1: Stuff Error

2: Form Error

3: Acknowledgment Error

4: Bit recessive Error

5: Bit dominant Error

6: CRC Error

7: Set by software

error_flag

bit0: Error warning flag

This bit is set by hardware when the warning limit has been reached (Receive Error Counter or Transmit Error Counter≥96).

bit1: Error passive flag

This bit is set by hardware when the Error Passive limit has been reached (Receive Error Counter or Transmit Error Counter>127).

bit2: Bus-off flag

This bit is set by hardware when it enters the bus-off state. The bus-off state is entered on TEC overflow, greater than 255

5.12.3.2 Android

IMC_CAN_MSG_OBJECT

class:

mrm.define.can.**IMC_CAN_MSG_OBJECT**

Fields:

Field Name	Type	Input/Output	Comment
can_bus_number	int	in	CAN channel number.
message_type	int	in, out	CAN message type.
id	long	in, out	CAN message identifier. For CAN 2.0A (11 bit identifiers), the valid value range is 0 to 7FFh For CAN 2.0B (29 bit identifiers), the valid value range is 0 to 1FFFFFFFh
buf	byte[]	in, out	Data of CAN message. This field must be a byte array of length of MAX_CAN_MESSAGE_BUFFER_SIZE(8)
buf_len	byte	in, out	Data length of message. When writing message, the value indicates the bytes to be written from "buf". The value must not greater then length of "buf".

IMC_CAN_MASK_OBJECT

class:

mrm.define.can.**IMC_CAN_MASK_OBJECT**

Fields:

Field Name	Type	Input/Output	Comment
can_bus_number	int	in	CAN channel number
mask_number	int	in	Filter ID
message_type	int	in, out	Target CAN message type to be applied
id	long	in, out	Target CAN message identifier value. For CAN 2.0A (11 bit identifiers), the valid value range is 0 to 7FFh For CAN 2.0B (29 bit identifiers), the valid value range is 0 to 1FFFFFFFh
mask	long	in, out	Mask value. Define the bits that MCU should care. The valid value range is 0 to FFFFFFFFh.

IMC_CAN_ERROR_STATUS_OBJECT

class:

mrm.define.can.**IMC_CAN_ERROR_STATUS_OBJECT**

Fields:

Field Name	Type	Input/Output	Comment
rec	int	out	Receive error counter. When the CAN controller receive the bus error, the counter will increase.
tec	int	out	Transmit error counter. When transmit CAN message encounter error, the counter will increase.
last_error_code	int	out	Define the last error code on bus 0: No Error 1: Stuff Error 2: Form Error 3: Acknowledgment Error 4: Bit recessive Error 5: Bit dominant Error 6: CRC Error 7: Set by software
error_flag	int	out	bit0: Error warning flag This bit is set by hardware when the warning limit has been reached (Receive Error Counter or Transmit Error Counter≥96). bit1: Error passive flag This bit is set by hardware when the Error Passive limit has been reached (Receive Error Counter or Transmit Error Counter>127). bit2: Bus-off flag This bit is set by hardware when it enters the bus-off state. The bus-off state is entered on TEC overflow, greater than 255

5.12.4 APIs

5.12.4.1 SUSI_IMC_CAN_Read

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_CAN_Read** (OUT PIMC_CAN_MSG_OBJECT object)
Android	short **SUSI_IMC_CAN_Read**(IMC_CAN_MSG_OBJECT canMsgObject)

Description:

Read a received CAN messages from the SDK internal buffer.

Parameters:

object [out]

Linux / Windows

Points to a variable that specifies the message of the CAN to be received.

Android

A instance of class **IMC_CAN_MSG_OBJECT** for storing the returned received CAN message.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

The default SDK internal buffer message queue can store 2048 CAN message. If application does not continuously consume the message from queue. The queue may overflow. SDK will remove oldest message to get the new one if there is no space in queue.

5.12.4.2 SUSI_IMC_CAN_Write

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_CAN_Write** (IN PIMC_CAN_MSG_OBJECT object)
Android	short **SUSI_IMC_CAN_Write**(IMC_CAN_MSG_OBJECT canMsgObject)

Description:

Write a message of CAN messages to the vehicle network.

Parameters:

object [in]

Linux / Windows

Points to a variable that specifies the message of the CAN message to be sent.

Android

A instance of class **IMC_CAN_MSG_OBJECT** for storing the CAN message to be sent

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

If you request the RTR message, you should defined the Buf_len=0 or you will get the undefined behavior

5.12.4.3 SUSI_IMC_CAN_GetBusErrorStatus

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_CAN_GetBusErrorStatus** (IN UINT can_bus_number, OUT PIMC_CAN_ERROR_STATUS_OBJECT object)
Android	short **SUSI_IMC_CAN_GetBusErrorStatus**(int canBusNumber, IMC_CAN_ERROR_STATUS_OBJECT status)

Description:

Get the bus error status.

Parameters:

can_bus_number [in]

CAN bus port number

IMC_CAN_MASK_OBJECT [out]

Windows:

Points to a variable that specifies the message mask of the CAN.

Android:

A instance of class **IMC_CAN_ERROR_STATUS_OBJECT** for storing the returned received CAN bus error status

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.12.4.4 SUSI_IMC_CAN_SetEvent

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_CAN_SetEvent** (IN void* Event)

Description:

Set event to listen CAN data from data bus. When VCIL receive a CAN message from bus, the event will be active. Event object is a HANDLE on Windows and the event object is pthread_cond_t on Linux.

Parameters:

hEvent [in]

Event handler of CAN message.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.12.4.5 SUSI_IMC_CAN_SetEventHandler

Syntax:

Android	short **SUSI_IMC_CAN_SetEventHandler**(Handler handler)

Description:

Set handler which handles CAN message received event.

Parameters:

handler [in]

An instance of Handler. Can not be null.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Remark:

- Please refer to the usage guide and sample code for for details.
- On alarm event triggered, the Handler will receive message with the "what" field equals to VCIL_EVENT_ID_RECEIVED_MSG_CAN

5.12.4.6 SUSI_IMC_CAN_UnsetEventHandler

Syntax:

Android	short **SUSI_IMC_CAN_UnsetEventHandler**()

Description:

Unregister handler of CAN message received event.

Parameters:

none

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Remark:

- Please refer to the usage guide and sample code for for details.

5.12.4.7 SUSI_IMC_CAN_WaitEvent

Syntax:

Android	short **SUSI_IMC_CAN_WaitEvent**(boolean status)

Description:

Allow/Disallow VCIL to pass CAN message received event to registered handler when a message is pushed into SDK internal buffer.

Parameters:

status [in]

The status of whether VCIL should pass CAN message received event to registered handler

TRUE: Inform

FLASE: Not to inform

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Remark:

- Please refer to the usage guide and sample for for details.

5.12.4.8 SUSI_IMC_CAN_SetBitTiming

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_CAN_SetBitTiming** (IN UINT can_bus_number, IN CAN_SPEED bit_rate)
Android	short **SUSI_IMC_CAN_SetBitTiming**(int canBusNumber, int bitRate)

Description:

Configure the CAN Bus speed to specific port.

Parameters:

can_bus_number [in]

Linux / Windows / Android

CAN bus number. The available value is 1 or 2.

bit_rate [in]

Linux / Windows:

Speed of can bus :

0: 125Kbps.

1: 250Kbps.

2: 500Kbps.

3: 1Mbps.

4: 200Kbps

5: 100Kbps

Android:

Speed of CAN bus.

Available values are enumerated in class CAN_SPEED

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.12.4.9 SUSI_IMC_CAN_SetBitTimingSilence

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_CAN_SetBitTimingSilence** (IN UINT can_bus_number, IN CAN_SPEED bit_rate)
Android	short **SUSI_IMC_CAN_SetBitTimingSilence**(int canBusNumber, int bitRate)

Description:

Configure the CAN Bus speed to specific port and set to operation mode on silence mode.

Parameters:

can_bus_number [in]

CAN bus number

bit_rate [in]

Speed of can bus :

0: 125Kbps.

1: 250Kbps.

2: 500Kbps.

3: 1Mbps.

4: 200Kbps

5: 100Kbps

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.12.4.10 SUSI_IMC_CAN_GetBitTiming

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_CAN_GetBitTiming** (IN UINT can_bus_number, OUT CAN_SPEED *bit_rate, OUT CAN_BUS_MODE *mode)
Android	short **SUSI_IMC_CAN_GetBitTiming**(int canBusNumber, int[] bitRate, int[] busMode)

Description:

Get currently the CAN Bus speed and operation mode to specific port.

Parameters:

can_bus_number [in]

Linux / Windows

CAN bus number. The available value is 1 or 2.

bit_rate [out]

Linux / Windows:

A pointer to the variable which stores the returned value of CAN bus bit rate.

Android:

An allocated array of size 1 for storing the returned value of CAN bus bit rate..

The return value will be stored at index 0.

Speed of can bus :

0: 125Kbps.

1: 250Kbps.

2: 500Kbps.

3: 1Mbps.

4: 200Kbps

5: 100Kbps

254: initiation (bus-off)

255: User define

mode [out]

Linux / Windows:

A pointer to the variable which stores the returned value of CAN bus mode.

For details, please refer to CAN_BUS_MODE

Android:

An allocated array of size 1 for storing the returned value of CAN bus mode.

The return value will be stored at index 0.

For details, please refer to CAN_BUS_MODE

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

This function only support the version above 2.5 firmware

TREK-520 Linux / CE

TREK-570 Linux / Windows

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.12.4.11 SUSI_IMC_CAN_SetMessageMask

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_CAN_SetMessageMask** (IN PIMC_CAN_MASK_OBJECT mask_object)
Android	short **SUSI_IMC_CAN_SetMessageMask**(IMC_CAN_MASK_OBJECT canMaskObject)

Description:

Add the message id with mask to acceptance filter mask.

Parameters:

IMC_CAN_MASK_OBJECT [in]

Linux / Windows

Points to a variable that specifies the message mask of the CAN to be added.

Android

A instance of class **IMC_CAN_MASK_OBJECT** for storing the CAN mask settings to be applied.

You should set the fields "can_bus_number" and "mask_number" to specified the target filter to be set.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

If application Set bank 0 message mask, the default mask rule will be covered.

5.12.4.12 SUSI_IMC_CAN_GetMessageMask

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_CAN_GetMessageMask** (OUT PIMC_CAN_MASK_OBJECT pmask_object)
Android	short **SUSI_IMC_CAN_GetMessageMask**(IMC_CAN_MASK_OBJECT canMaskObject)

Description:

Get massage filter mask.

Parameters:

IMC_CAN_MASK_OBJECT [out]

Linux / Windows

Points to a variable that specifies the message mask of the CAN.

Android

A instance of class **IMC_CAN_MASK_OBJECT** for storing the returned CAN mask settings. You should set the fields "can_bus_number" and "mask_number" to specified the target filter setting to be queried.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

When you want to use this API, you only need to specify the can_bus_number and mask_number.

5.12.4.13 SUSI_IMC_CAN_RemoveMessageMask

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_CAN_RemoveMessageMask** (IN UINT can_bus_number, IN int mask_number)
Android	short **SUSI_IMC_CAN_RemoveMessageMask**(int canBusNumber, int maskNumber)

Description:

Remove message filter mask.

Parameters:

can_bus_number [in]

Linux / Windows / Android

CAN bus number.

mask_number [in]

Linux / Windows / Android

Message filter ID.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

If application remove bank 0 message mask, the default mask rule will be remove. All message may not receive when another message mask unset.

5.12.4.14 SUSI_IMC_CAN_ResetMessageMask

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_CAN_ResetMessageMask** (IN UINT can_bus_number)
Android	short **SUSI_IMC_CAN_ResetMessageMask**(int canBusNumber)

Description:

Reset all message filter masks to default.

Parameters:

can_bus_number [in]

Linux / Windows / Android

CAN bus number.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

Reset the message filter. The default mask add to bank 0 with mask equal 0 and all message will be received.

5.13 J1939

5.13.1 Usage Overview

TREK device supports data transmission of J1939 protocol and provides filter functions.
The SDK hold a internal buffer for received J1939 messages with limited size. You can use the read API to consume the buffer and get received message. When the internal buffer become full before application consume it, the oldest message will be removed and new message will be added.

The following section describe the usage of corresponding APIs.

5.13.1.1 Basic Usage

Windows/Linux

To start using J1939 APIs you should first initialize the APIs and activate J1939 protocol on specific CAN channel.

Please refer to the VCIL usage for how to initialize and configure CAN channels.

Android

The class for J1939 APIs for Android is **mrm.VCIL**
To start using J1939 APIs you should first initialize the APIs and activate J1939 protocol on specific CAN channel .

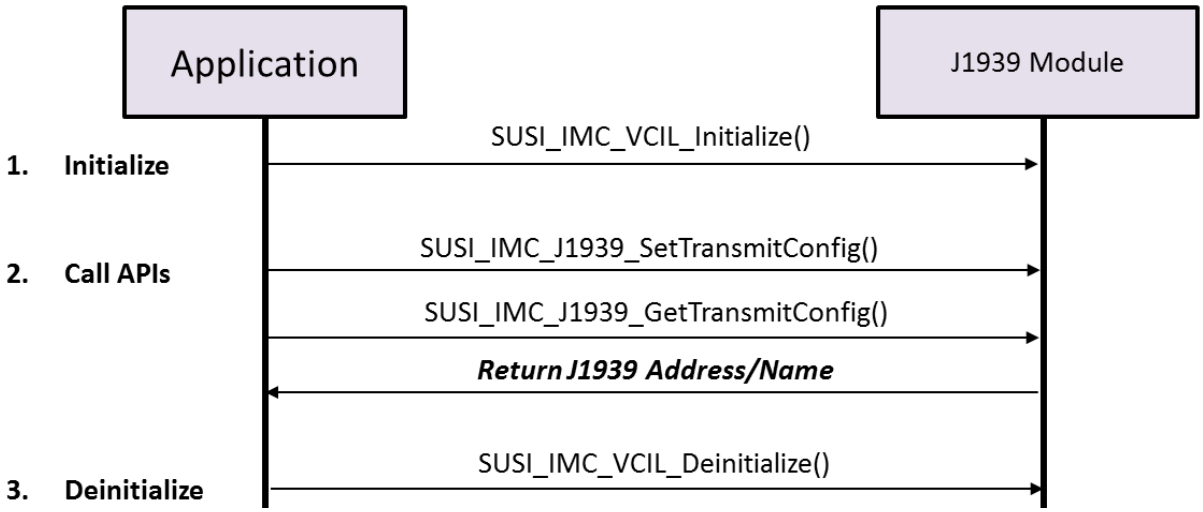
Please refer to the VCIL usage for basic usage and for how to initialize and configure CAN channels.

5.13.1.2 J1939 Transmit Config Settings

To set the J1939 Address and Name, you can use SUSI_IMC_J1939_SetTransmitConfig() . To get current config you can use SUSI_IMC_J1939_GetTransmitConfig() .

The rule of J1939 Address and Name follows SAE_J1939/81. Please refer to [SAE J1939/81].

Windows/ Linux/ Android



5.13.1.3 CAN Bus Speed Setting

To start data transmission of CAN, J1939, OBD2 protocol on CAN bus, you must first configure the CAN bus speed for MCU.

Please refer to the section CAN Bus Speed Setting.

5.13.1.4 J1939 Message Reading

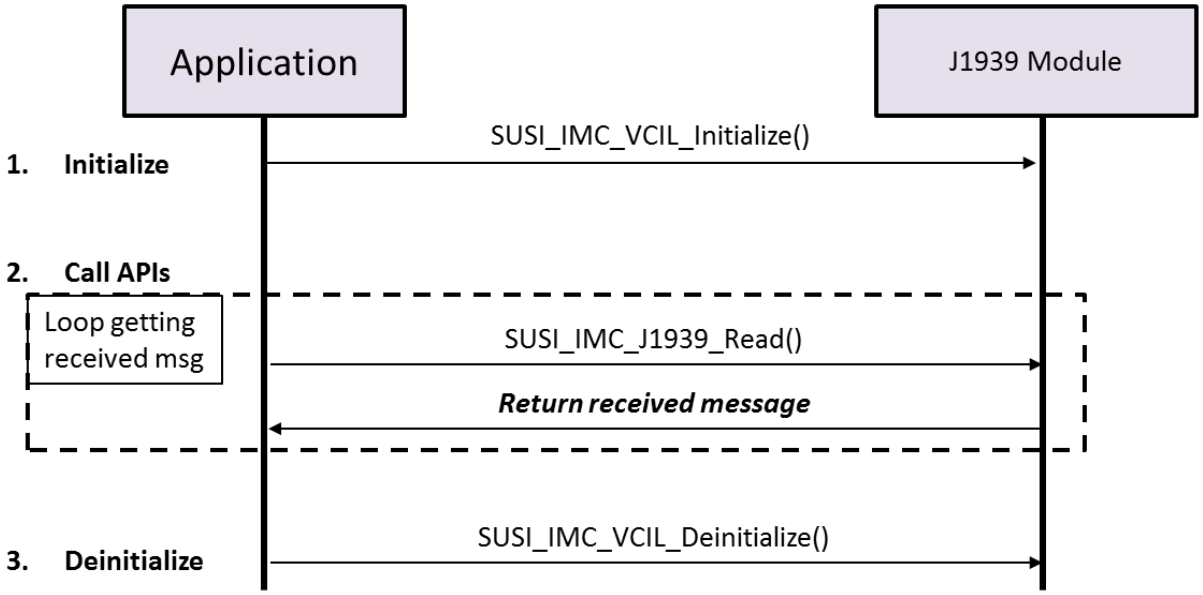
The CAN bus speed must be set correctly before you do read/write operation. Please refer to CAN Bus Speed Setting section for the details.

The mechanism of reading J1939 messages from CAN bus by using SDK APIs is basically the same with CAN. Please refer to the section CAN Message Reading . This section will only shows the brief flow of using J1939 APIs.

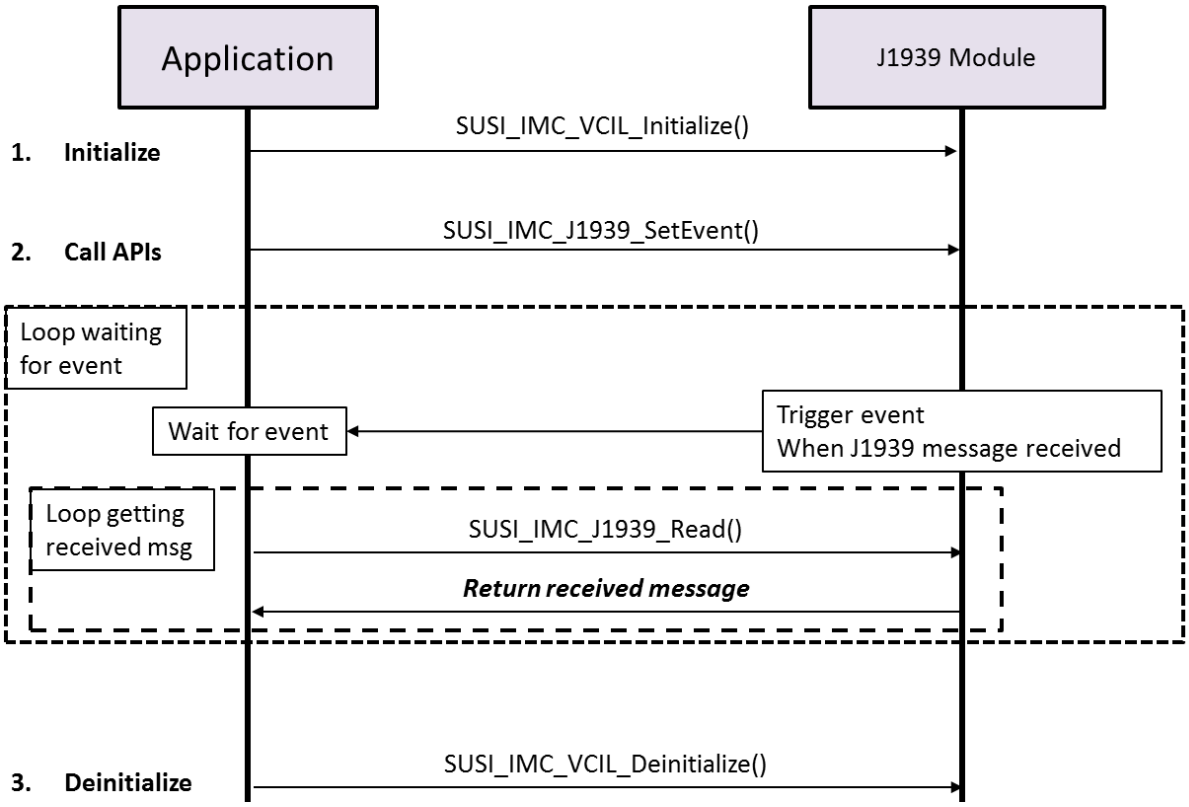
Windows/Linux

You implement your J1939 message reading application in either **Polling** or **Event Handling** style.

- **Polling:**



- **Event Handling:**

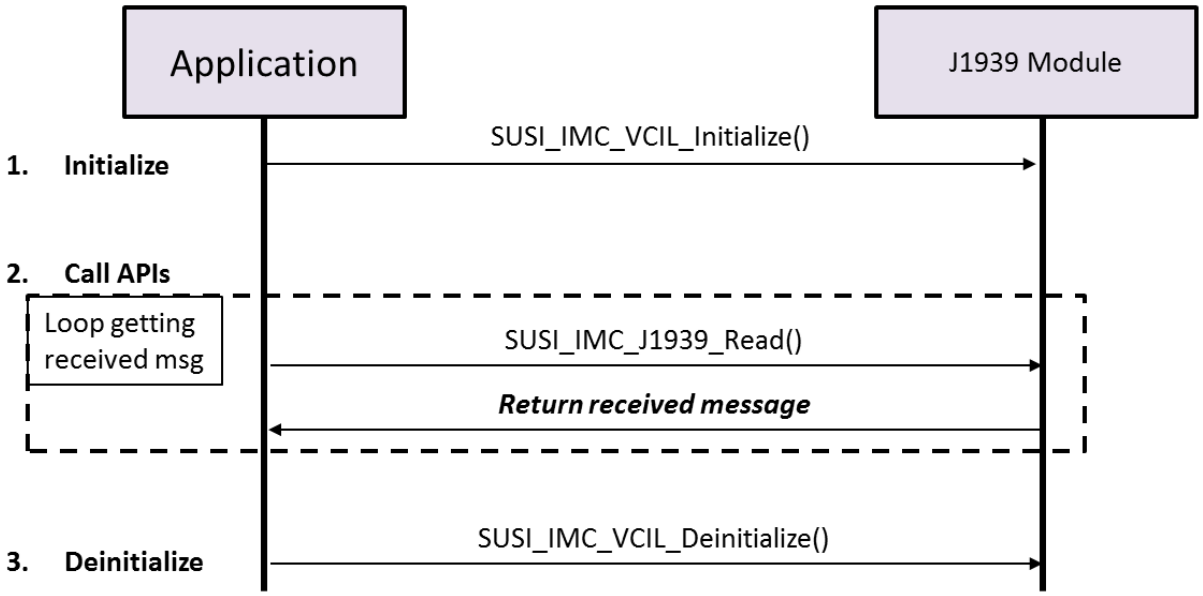

Application
J1939 Module
1. Initialize
SUSI_IMC_VCIL_Initialize()
2. Call APIs
SUSI_IMC_J1939_SetEvent()
Loop waiting for event
Wait for event
Trigger event
When J1939 message received
Loop getting received msg
SUSI_IMC_J1939_Read()
Return received message
3. Deinitialize
SUSI_IMC_VCIL_Deinitialize()

Android

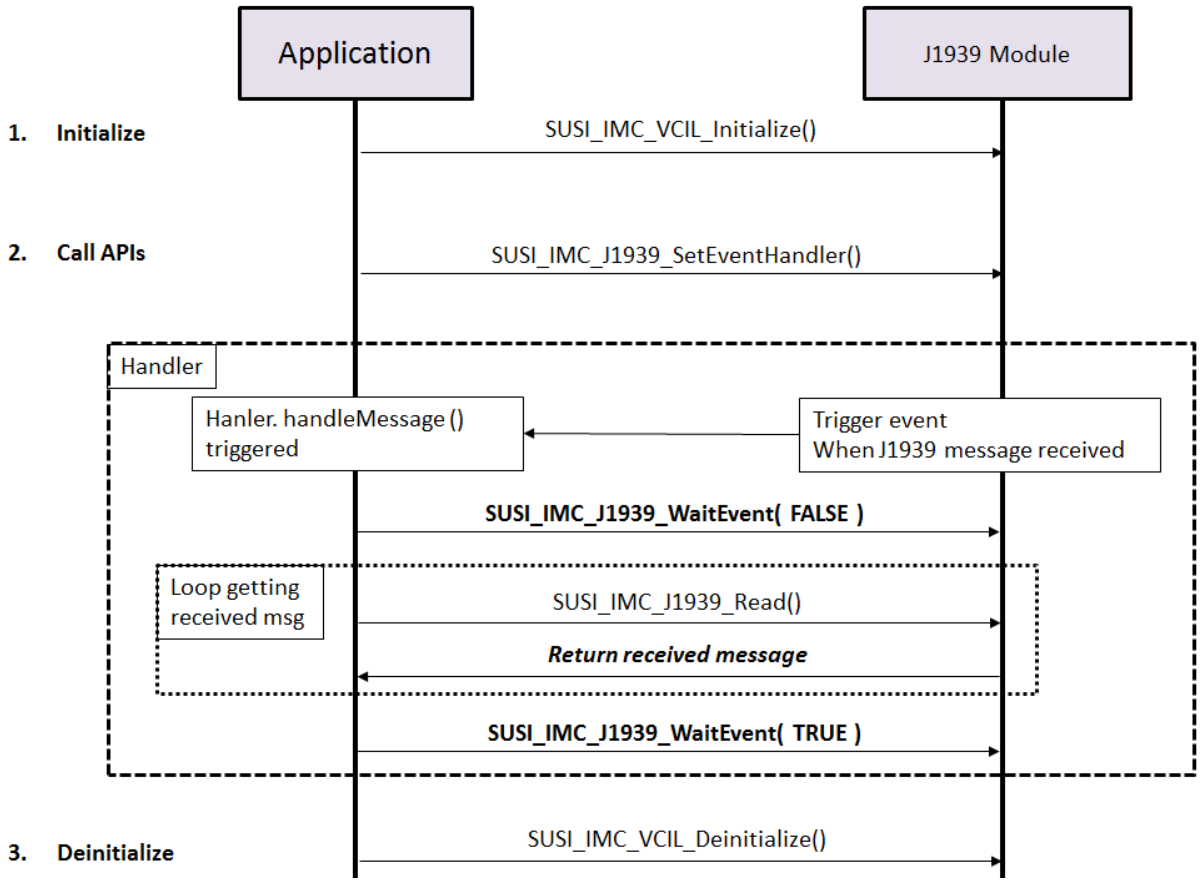
For the details of usage, please also refer to CAN Message Reading for Android.

You implement your J1939 message reading application in either **Polling** or **Event Handling** style.

- **Polling:**



- **Event Handling:**

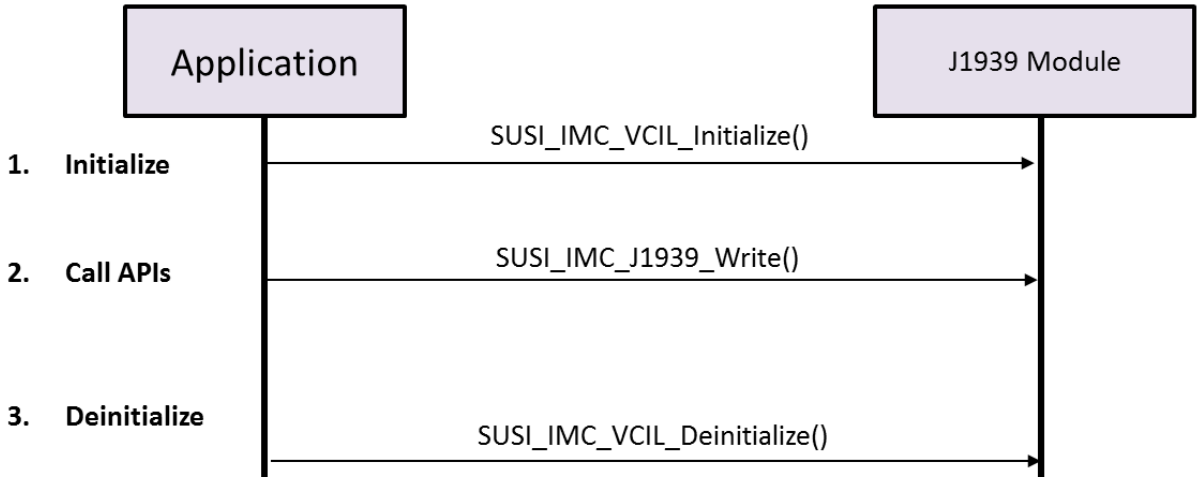

Application
J1939 Module
1. Initialize
SUSI_IMC_VCIL_Initialize()
2. Call APIs
SUSI_IMC_J1939_SetEventHandler()
Handler
Hanler. handleMessage () triggered
Trigger event When J1939 message received
SUSI_IMC_J1939_WaitEvent(FALSE)
Loop getting received msg
SUSI_IMC_J1939_Read()
Return received message
SUSI_IMC_J1939_WaitEvent(TRUE)
3. Deinitialize
SUSI_IMC_VCIL_Deinitialize()

5.13.1.5 J1939 Message Writing

This section describe how to write J1939 messages to CAN bus by using SDK APIs.

The CAN bus speed must be set correctly before you do read/write operation. Please refer to CAN Bus Speed Setting section for the details.

Windows/ Linux/ Android



To write J1939 message to CAN bus, you can call SUSI_IMC_J1939_Write() with a IMC_J1939_MSG_OBJECT structure which contains the data you want to write.

5.13.1.6 J1939 Acceptance filter Settings

Similar with CAN Acceptance Filter, the VCIL MCU provide acceptance filter for J1939 protocol, which can filter the PGN of J1939 message. The VCIL MCU supports at most 128 PGN filters. A J1939 message with PGN which is matched with any of the filters will be accepted by MCU.

To enable the filter function, you **must configure the CAN physical channel to VCIL_MODE_J1939_WITH_MASK, to work with filter first**. Please refer to the Basic Usage of VCIL section.

The following is an example of filter setting. If we set filter with PGN: 0xFEF6, 0xFE20, 0xF201. When the received J1939 message's PGN value is 0xFEF6, the messages is accepted.

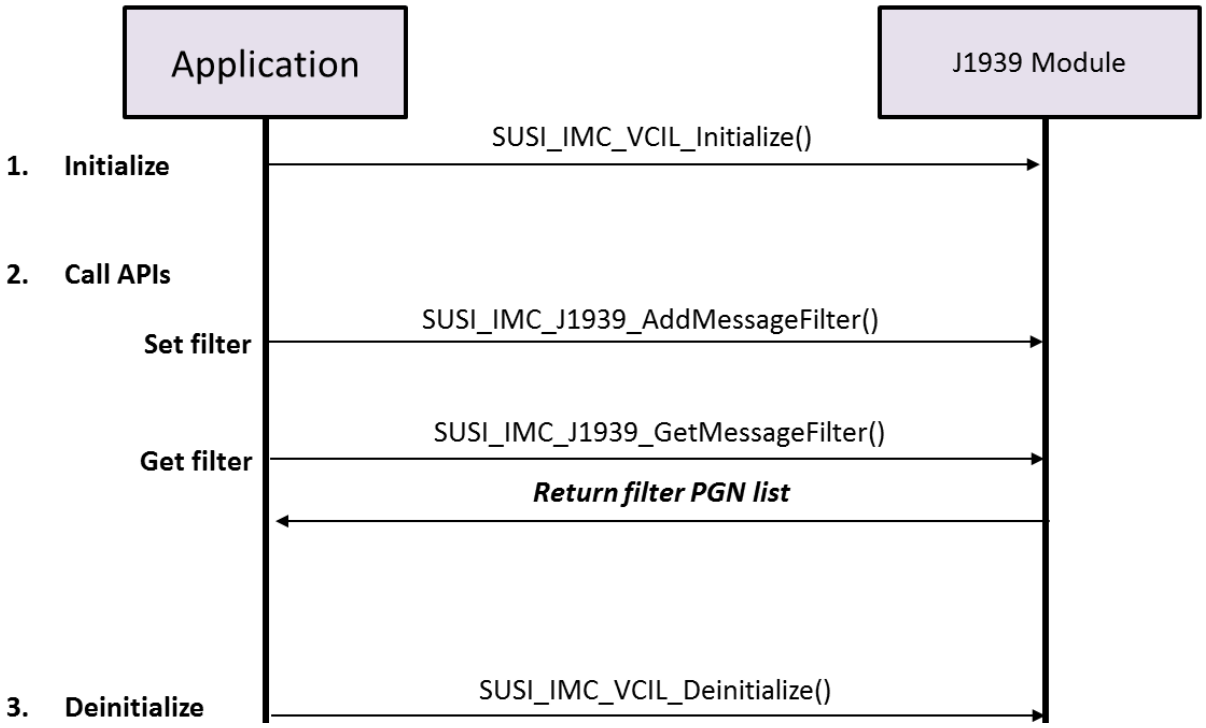
J1939 Filter Configuration	Receive J1939 data's PGN value	Result
Filter PGN=0xFEF6	0xFEF6	**Accept**
Filter PGN=0xFEF6	0xFEF7	**Drop**

J1939 Filter Configuration	Receive J1939 data's PGN value	Result
Filter PGN=0xFEF6 OR 0xFE20 OR 0xF201	0xFEF6	**Accept**
Filter PGN=0xFEF6 OR 0xFE20 OR 0xF201	0xF201	**Accept**
Filter PGN=0xFEF6 OR 0xFE20 OR 0xF201	0xF203	**Drop**
Filter PGN=0xFEF6 OR 0xFE20 OR 0xF201	0xFE22	**Drop**

Windows/ Linux/ Android

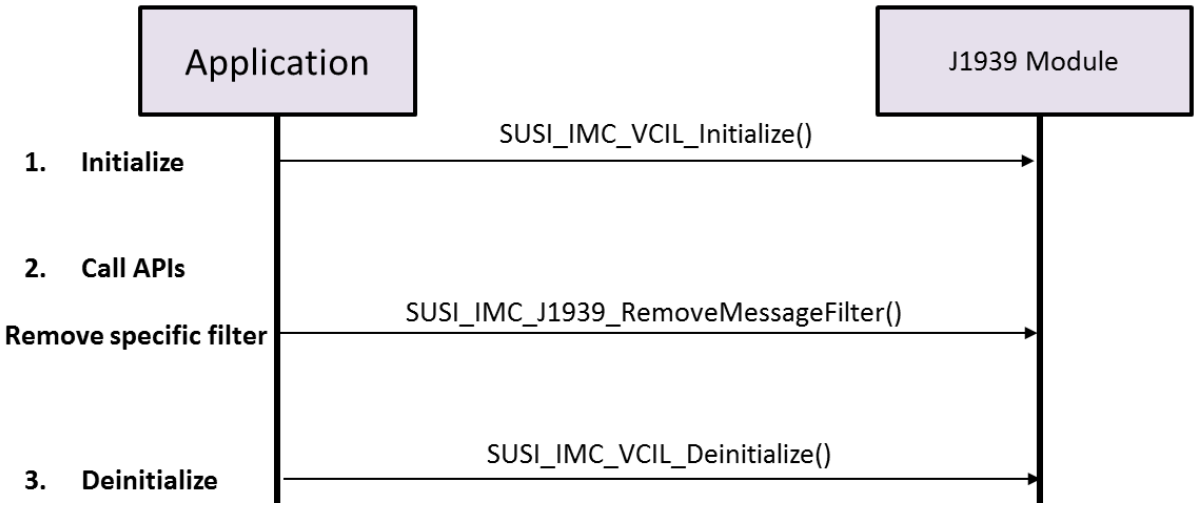
- **To get/set filter:**

Call SUSI_IMC_J1939_AddMessageFilter() to set the mask and call SUSI_IMC_J1939_GetMessageFilter() to get mask.



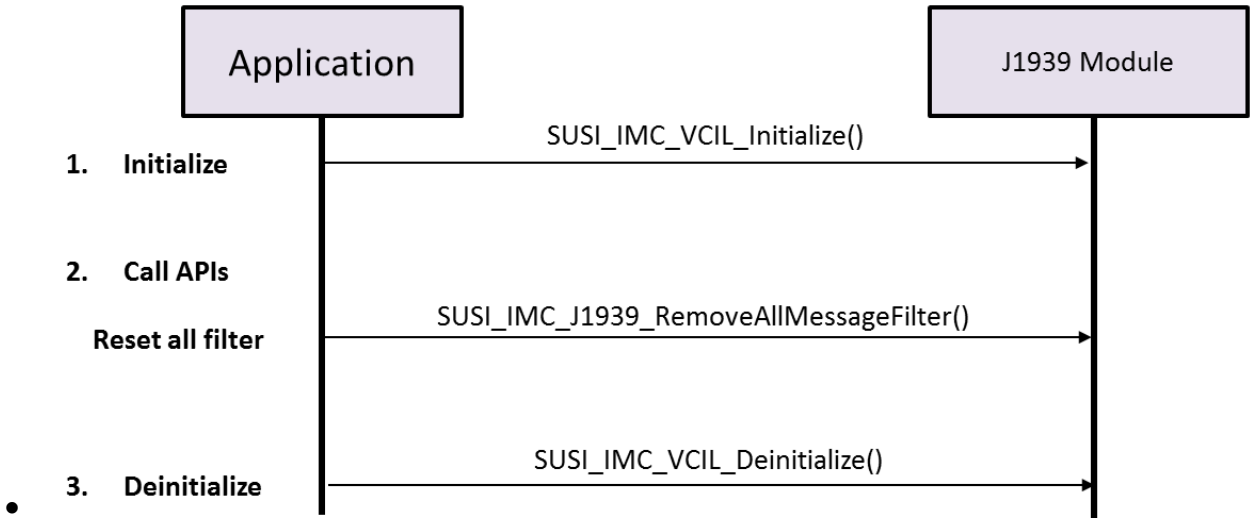
- **To remove specific filter:**

Call SUSI_IMC_J1939_RemoveMessageFilter() to remove filter of specified filter ID.



- **To remove all filter:**

Call SUSI_IMC_J1939_RemoveAllMessageFilter() to reset all filter.



5.13.2 Constant

5.13.2.1 Android

Class:
mrm.define.**IMC_CONSTANT**

Fields:

Field Name	Type	Value
MAX_J1939_MESSAGE_BUFFER_SIZE	int	64
MAX_J1939_FILTER_NUMBER	int	128
J1939_TRANSMIT_CONFIG_NAME_LEN	int	8

5.13.3 Structure/Class

5.13.3.1 Windows/Linux

IMC_J1939_MSG_OBJECT Structure

Syntax:

```
typedef struct
{
        BYTE channel_number;
            UINT pgn;
            BYTE dst;
            BYTE src;
            BYTE pri;
            BYTE buf[64];
            BYTE buf_len;
}IMC_J1939_MSG_OBJECT, *PIMC_J1939_MSG_OBJECT;
```

Description:

This data structure defines the J1939 message.

Members:

channel_number

Physical channel number.The range is 1 to 2.

pgn

Parameter group number.

dst

Destination of message.

src

Source of message.

pri

Priority of message.

buf

Point to a receive buffer.

Buf_len

Data length code of the message (range from 0 to 64).

IMC_J1939_TRANSMIT_CONFIG Structure

Syntax:

```
typedef struct
{
        BYTE channel_number;
        BYTE source_address;
        BYTE source_name[8];
}IMC_J1939_TRANSMIT_CONFIG, *PIMC_J1939_TRANSMIT_CONFIG;
```

Description:

This structure defines the J1939 transmit configuration.

Members:

channel_number

Physical channel number. The range is 1 to 2.

source_address

Specified source address.

Source_name

Base on J1939-81, Specified source NAME.

5.13.3.2 Android

IMC_J1939_MSG_OBJECT

class:

mrm.define.j1939.**IMC_J1939_MSG_OBJECT**

Fields:

Field Name	Type	Input/Output	Comment
channel_number	byte	in	CAN channel number. The range is 1 to 2.
pgn	int	in, out	parameter group number
dst	byte	in, out	destination of message.
src	byte	in, out	source of message
pri	byte	in	priority of message.
buf	byte[]	in, out	Data of message. This field must be a byte array of length of MAX_J1939_MESSAGE_BUFFER_SIZE(64)
buf_len	int	in, out	Data length of message. When writing message, the value indicates the bytes to be written from "buf". The value must not greater then length of "buf".

IMC_J1939_TRANSMIT_CONFIG

class:

mrm.define.j1939.**IMC_J1939_TRANSMIT_CONFIG**

Fields:

Field Name	Type	Input/Output	Comment
channel_number	byte	in	CAN channel number. The range is 1 to 2.
source_address	byte	in	Source address
source_name	byte[]	in, out	Base on J1939-81, Specified source NAME. This field must be a byte array of length of J1939_TRANSMIT_CONFIG_NAME_LEN(8)

5.13.4 APIs

5.13.4.1 SUSI_IMC_J1939_Read

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_J1939_Read** (OUT PIMC_J1939_MSG_OBJECT object);
Android	short **SUSI_IMC_J1939_Read** (IMC_J1939_MSG_OBJECT j1939MsgObject)

Description:

Read a J1939 messages from the library queue.

Parameters:

object [out]

Linux / Windows

Points to a variable that specifies the message of the J1939 to be received.

Android

A instance of class **IMC_J1939_MSG_OBJECT** for storing the returned received J1939 message

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

The default library message queue can store 2048 J1939 message. If application not continue query the message from queue. The queue may overflow when the J1939 bus data incoming continues. The J1939 message may overwrite the last message when the queue with Insufficient space.

5.13.4.2 SUSI_IMC_J1939_Write

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_J1939_Write** (IN PIMC_J1939_MSG_OBJECT object);
Android	short **SUSI_IMC_J1939_Write** (IMC_J1939_MSG_OBJECT j1939MsgObject)

Description:

Write a message of J1939 messages to the vehicle network.

Parameters:

object [out]

Linux / Windows

Points to a variable that specifies the message of the J1939 to be sent.

Android

A instance of class **IMC_J1939_MSG_OBJECT** for storing the J1939 message to be sent

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

The **src** field will ignore in the function. The actual message source address base on the **SUSI_IMC_J1939_SetTransmitConfig** address setting

5.13.4.3 SUSI_IMC_J1939_SetEvent

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_J1939_SetEvent**(void* hEvent);

Description:

Set event to listen J1939 data from bus. When VCIL receive a J1939 message from bus, the event will be active. Event object is a HANDLE on Windows and the event object is pthread_cond_t on Linux.

Parameters:

hEvent [in]

Event handler of J1939 message.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.13.4.4 SUSI_IMC_J1939_SetEventHandler

Syntax:

Android	short **SUSI_IMC_J1939_SetEventHandler**(Handler handler)

Description:

Set handler which handles J1939 message received event.

Parameters:

handler [in]

An instance of Handler. Can not be null.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Remark:

- Please refer to the usage guide and sample code for for details.
- On alarm event triggered, the Handler will receive message with the "what" field equals to VCIL_EVENT_ID_RECEIVED_MSG_J1939

5.13.4.5 SUSI_IMC_J1939_UnsetEventHandler

Syntax:

Android	short **SUSI_IMC_J1939_UnsetEventHandler**()

Description:

Unregister handler of J1939 message received event.

Parameters:

none

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Remark:

- Please refer to the usage guide and sample code for for details.

5.13.4.6 SUSI_IMC_J1939_WaitEvent

Syntax:

Android	short **SUSI_IMC_J1939_WaitEvent**(boolean status)

Description:

Allow/Disallow VCIL to pass J1939 message received event to registered handler when a message is pushed into SDK internal buffer.

Parameters:

status [in]

The status of whether VCIL should pass J1939 message received event to registered handler

TRUE: Inform

FLASE: Not to inform

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Remark:

- Please refer to the usage guide and sample for for details.

5.13.4.7 SUSI_IMC_J1939_AddMessageFilter

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_J1939_AddMessageFilter**(IN BYTE channel_number, UINT pgn);
Android	short **SUSI_IMC_J1939_AddMessageFilter**(byte channelNumber, int pgn)

Description:

Add the PGN to acceptance filter.

Parameters:

channel_number [in]

Linux / Windows / Android

Physical channel number. The range is 1 to 2.

pgn [in]

Linux / Windows / Android

PGN of filter to be added.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

The maximum support 128 sets pgn filter

5.13.4.8 SUSI_IMC_J1939_GetMessageFilter

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_J1939_GetMessageFilter**(IN BYTE channel_number, OUT UINT* total, UINT* pgn);
Android	short **SUSI_IMC_J1939_GetMessageFilter**(byte channelNumber, int[] total, int[] pgn)

Description:

Get massage pgn filters.

Parameters:

channel_number [in]

Linux / Windows / Android

Physical channel number. The range is 1 to 2.

total [out]

Linux / Windows

Points to a variable that specifies the total number of PGN filter.

Android

An allocated array of size 1 for storing the returned total number of PGN filters.

The return value will be stored at index 0.

pgn [out]

Linux / Windows

Points to a variable that specifies the message filters of the PGN.

Android

An allocated array of size MAX_J1939_FILTER_NUMBER (128) for storing the returned list of PGN filters.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

For Windows/Linux, this API must to call 2 times. First, user must to assign the null to mid. It will return

total number of mids. Second, user must to assign the array of pgns with total number. After second step, this API will return all message filters.

5.13.4.9 SUSI_IMC_J1939_RemoveMessageFilter

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_J1939_RemoveMessageFilter**(IN BYTE channel_number, UINT pgn);
Android	short **SUSI_IMC_J1939_RemoveMessageFilter**(byte channelNumber, int pgn)

Description:

Remove message filter.

Parameters:

channel_number [in]

Linux / Windows / Android

Physical channel number. The range is 1 to 2.

pgn [in]

Linux / Windows / Android

PGN of filter to be removed

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.13.4.10 SUSI_IMC_J1939_RemoveAllMessageFilter

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_J1939_RemoveAllMessageFilter** (IN BYTE channel_number);
Android	short **SUSI_IMC_J1939_RemoveAllMessageFilter**(byte channelNumber)

Description:

Reset all specified physical channel filters to the default value.

Parameters:

channel_number [in]

Linux / Windows / Android

Physical channel number. The range is 1 to 2.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.13.4.11 SUSI_IMC_J1939_SetTransmitConfig

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_J1939_SetTransmitConfig**(IN IMC_J1939_TRANSMIT_CONFIG transmit_config);
Android	short **SUSI_IMC_J1939_SetTransmitConfig**(IMC_J1939_TRANSMIT_CONFIG j1939TransmitConfig)

Description:

Set address and NAME of the J1939 device.

Parameters:

transmit_config [in]

Linux / Windows

Points to a variable that specifies the address and name of the J1939 device.

Android

A instance of class **IMC_J1939_TRANSMIT_CONFIG** for storing config values to be set

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

When this API calling. The device will claiming Address and NAME to the bus. If the address occupied the address will auto shift to next available address.

5.13.4.12 SUSI_IMC_J1939_GetTransmitConfig

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_J1939_GetTransmitConfig**(OUT PIMC_J1939_TRANSMIT_CONFIG ptransmit_config);
Android	short **SUSI_IMC_J1939_GetTransmitConfig**(IMC_J1939_TRANSMIT_CONFIG j1939TransmitConfig)

Description:

Get address and NAME of the J1939 device.

Parameters:

transmit_config [out]

Linux / Windows

Points to a variable that specifies the address and name of the J1939 device.

Android

A instance of class **IMC_J1939_TRANSMIT_CONFIG** for storing returned config values.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.14 OBD2

5.14.1 Usage Overview

TREK device supports data transmission of OBD2 protocol and provides filter functions.
The SDK hold a internal buffer for received OBD2 messages with limited size. You can use the read API to consume the buffer and get received message. When the internal buffer become full before application consume it, the oldest message will be removed and new message will be added.

The following section describe the usage of corresponding APIs.

5.14.1.1 Basic Usage

Windows/Linux

To start using OBD2 APIs you should first initialize the APIs and activate OBD2 protocol on specific CAN channel.

Please refer to the VCIL usage for how to initialize and configure CAN channels.

Android

The class for OBD2 APIs for Android is **mrm.VCIL**
To start using OBD2 APIs you should first initialize the APIs and activate OBD2 protocol on specific CAN channel .

Please refer to the VCIL usage for basic usage and for how to initialize and configure CAN channels.

5.14.1.2 CAN Bus Speed Setting

To start data transmission of CAN, J1939, OBD2 protocol on CAN bus, you must first configure the CAN bus speed for MCU.

Please refer to the section CAN Bus Speed Setting.

5.14.1.3 OBD2 Message Reading

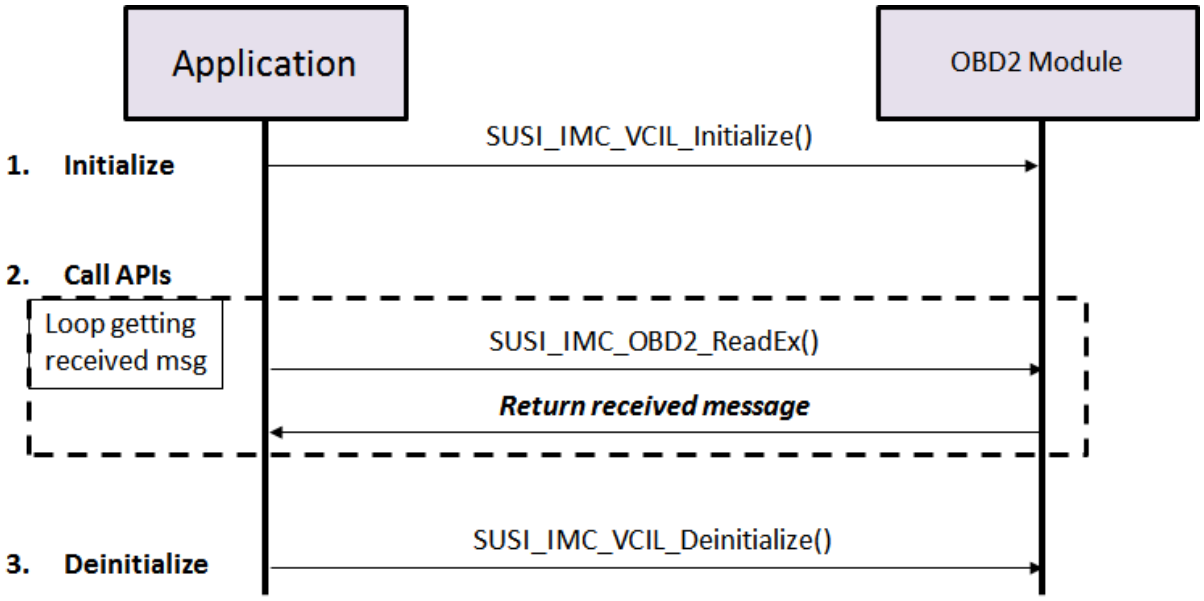
The CAN bus speed must be set correctly before you do read/write operation. Please refer to CAN Bus Speed Setting section for the details.

The mechanism of reading OBD2 messages from CAN bus by using SDK APIs is basically the same with CAN. Please refer to the section CAN Message Reading . This section will only shows the brief flow of using OBD2 APIs.

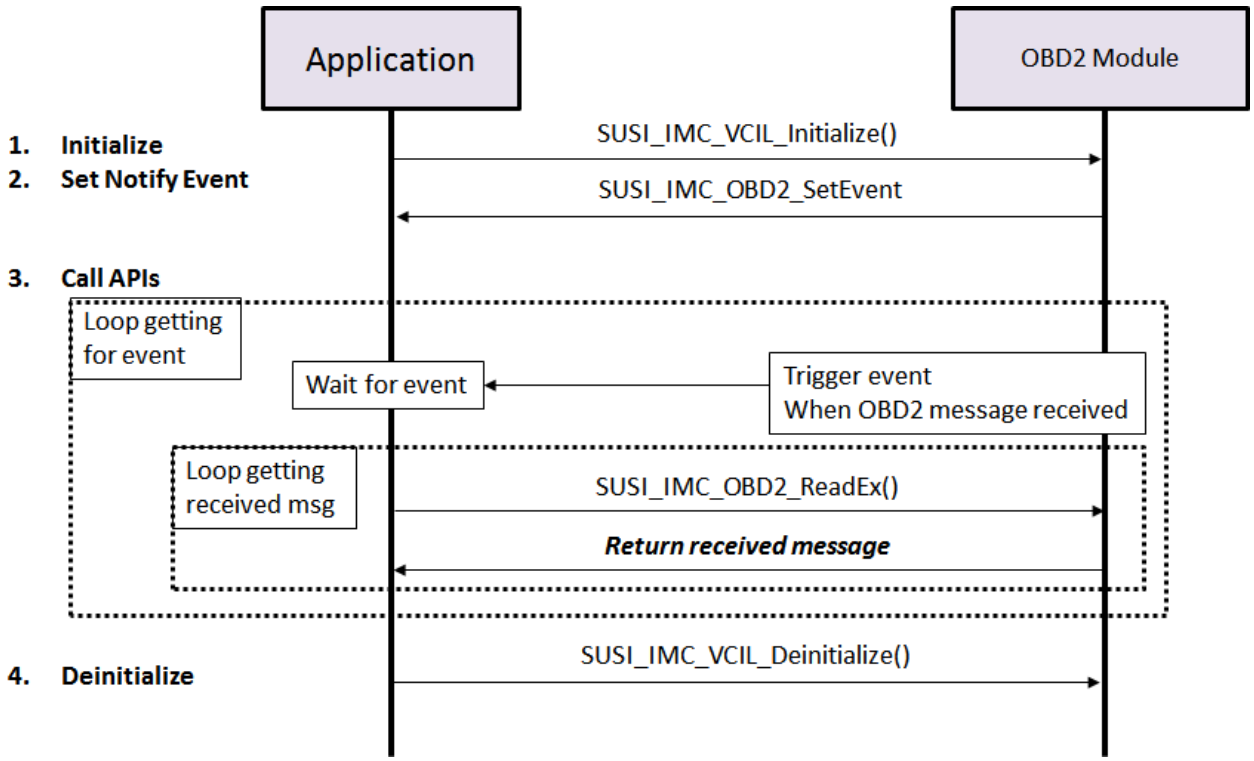
Windows/Linux

You implement your OBD2 message reading application in either **Polling** or **Event Handling** style.

- **Polling:**



- **Event Handling:**

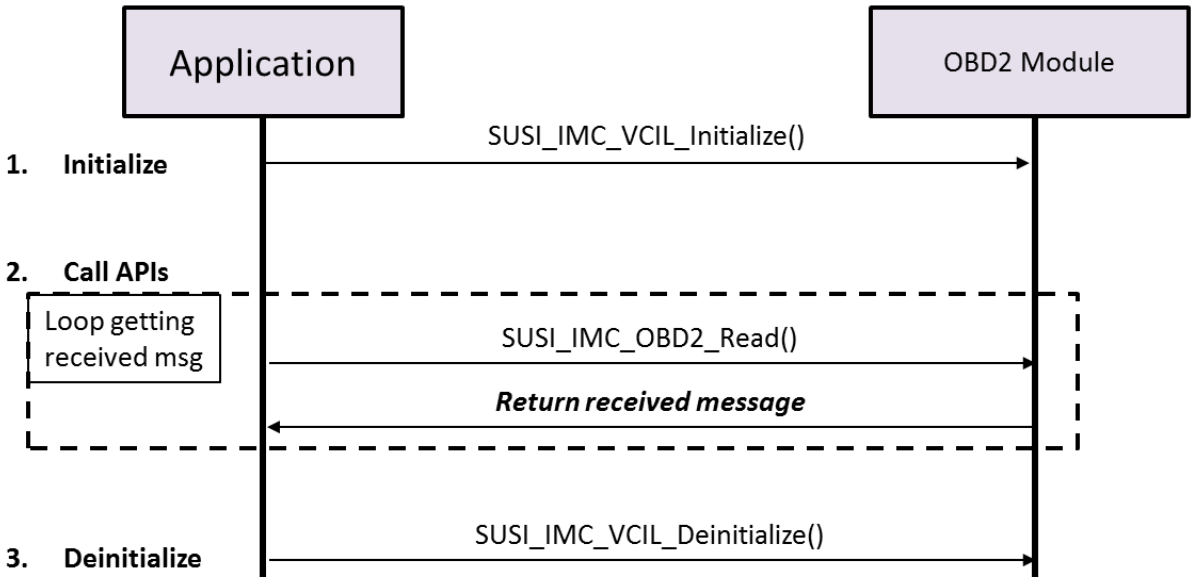

Application
OBD2 Module
1. Initialize
2. Set Notify Event
SUSI_IMC_VCIL_Initialize()
SUSI_IMC_OBD2_SetEvent
3. Call APIs
Loop getting for event
Wait for event
Trigger event When OBD2 message received
Loop getting received msg
SUSI_IMC_OBD2_ReadEx()
Return received message
4. Deinitialize
SUSI_IMC_VCIL_Deinitialize()

Android

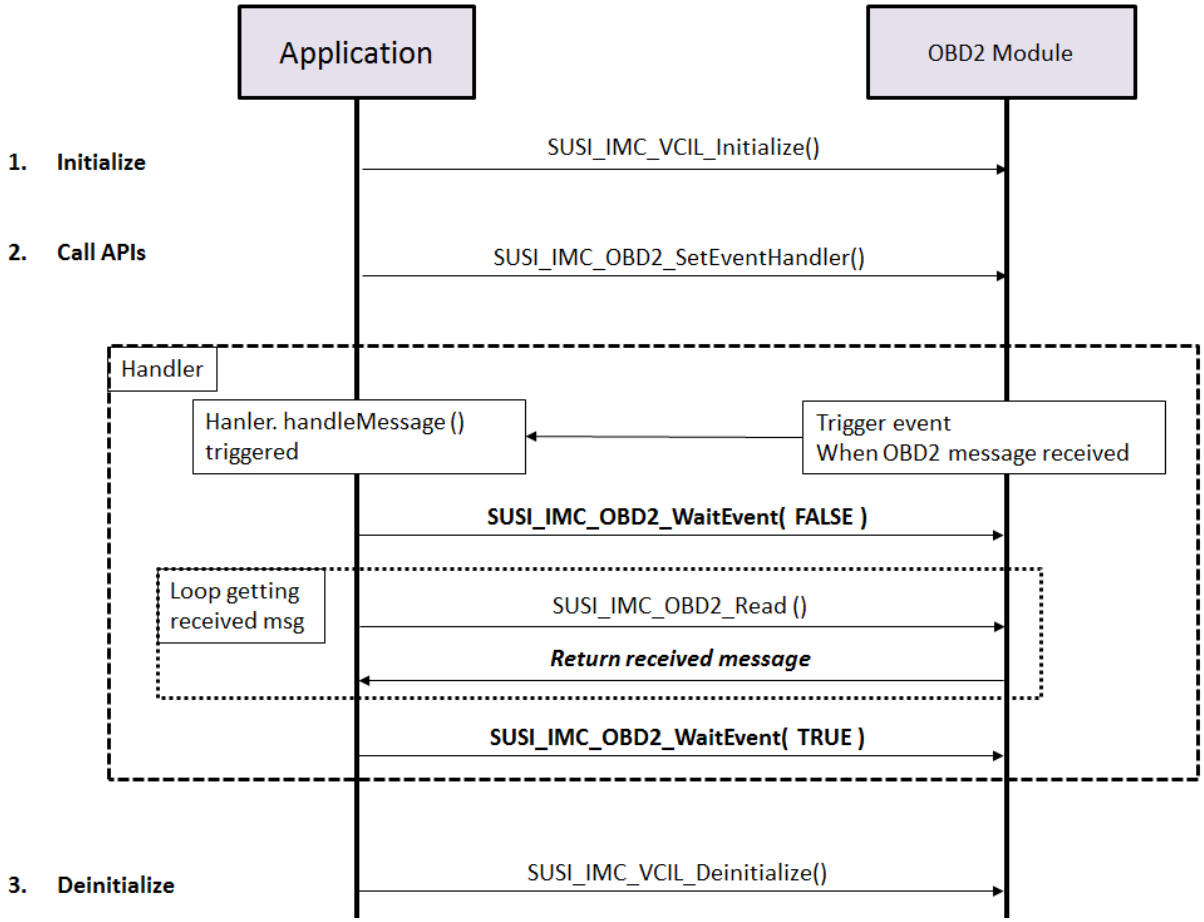
For the details of usage, please also refer to [CAN Message Reading for Android](#).

You implement your OBD2 message reading application in either **Polling** or **Event Handling** style.

- **Polling:**



- **Event Handling:**

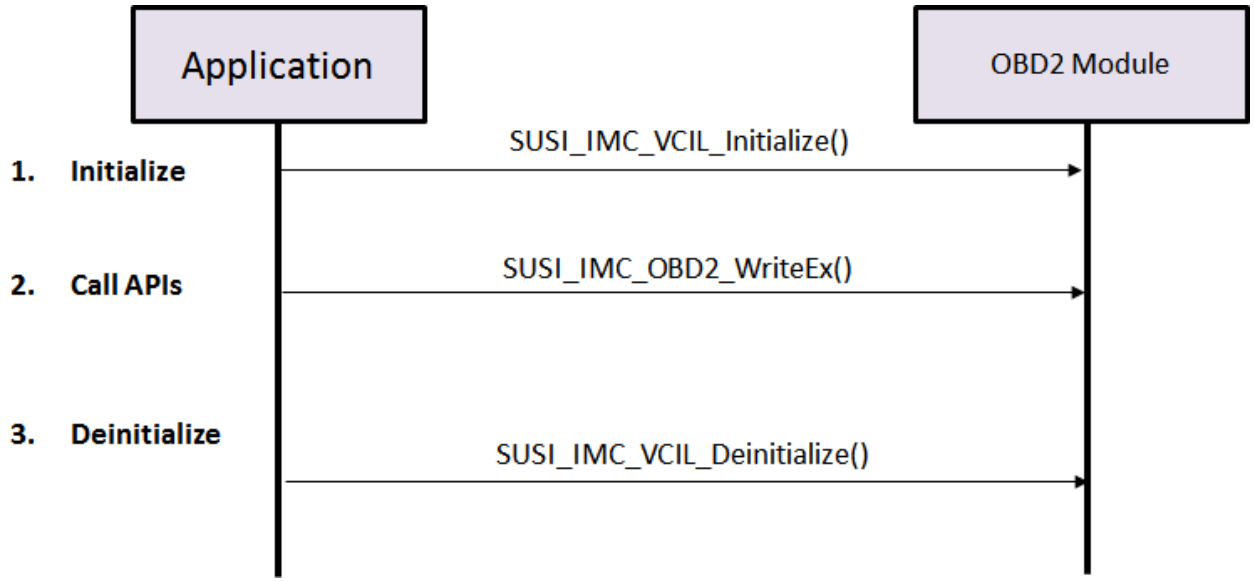

Application
OBD2 Module
1. Initialize
SUSI_IMC_VCIL_Initialize()
2. Call APIs
SUSI_IMC_OBD2_SetEventHandler()
Handler
Hanler. handleMessage () triggered
Trigger event When OBD2 message received
SUSI_IMC_OBD2_WaitEvent(FALSE)
Loop getting received msg
SUSI_IMC_OBD2_Read ()
Return received message
SUSI_IMC_OBD2_WaitEvent(TRUE)
3. Deinitialize
SUSI_IMC_VCIL_Deinitialize()

5.14.1.4 OBD2 Message Writing

This section describe how to write OBD2 messages to CAN bus by using SDK APIs.

The CAN bus speed must be set correctly before you do read/write operation. Please refer to CAN Bus Speed Setting section for the details.

Windows/ Linux/ Android



To write OBD2 message to CAN bus, you can call SUSI_IMC_OBD2_WriteEx() with a IMC_OBD2_MSG_EX_OBJECT structure which contains the data you want to write.

5.14.1.5 OBD2 Acceptance filter Settings

Similar with CAN Acceptance Filter, the VCIL MCU provide acceptance filter for OBD2 protocol, which can filter the PID of OBD2 message. The VCIL MCU supports at most 256 PID filters. A OBD2 message with PID which is matched with any of the filters will be accepted by MCU.

To enable the filter function, you **must configure the CAN physical channel to VCIL_MODE_OBD2_WITH_MASK, to work with filter first**. Please refer to the Basic Usage of VCIL section.

The following is an example of filter setting. If we set filter with PID 0x80, 0x56, 0x11 When the received OBD2 message's PID value is 0x56, the messages is accepted.

OBD2 Filter Configuration	Receive OBD2 data's PID value	Result
Filter PID=0x80	0x80	**Accept**
Filter PID=0x80	0x79	**Drop**

OBD2 Filter Configuration	Receive OBD2 data's PID value	Result
Filter PID=0x80 OR 0x56 OR 0x11	0x80	**Accept**
Filter PID=0x80 OR 0x56 OR 0x11	0x56	**Accept**
Filter PID=0x80 OR 0x56 OR 0x11	0x57	**Drop**
Filter PID=0x80 OR 0x56 OR 0x11	0x12	**Drop**

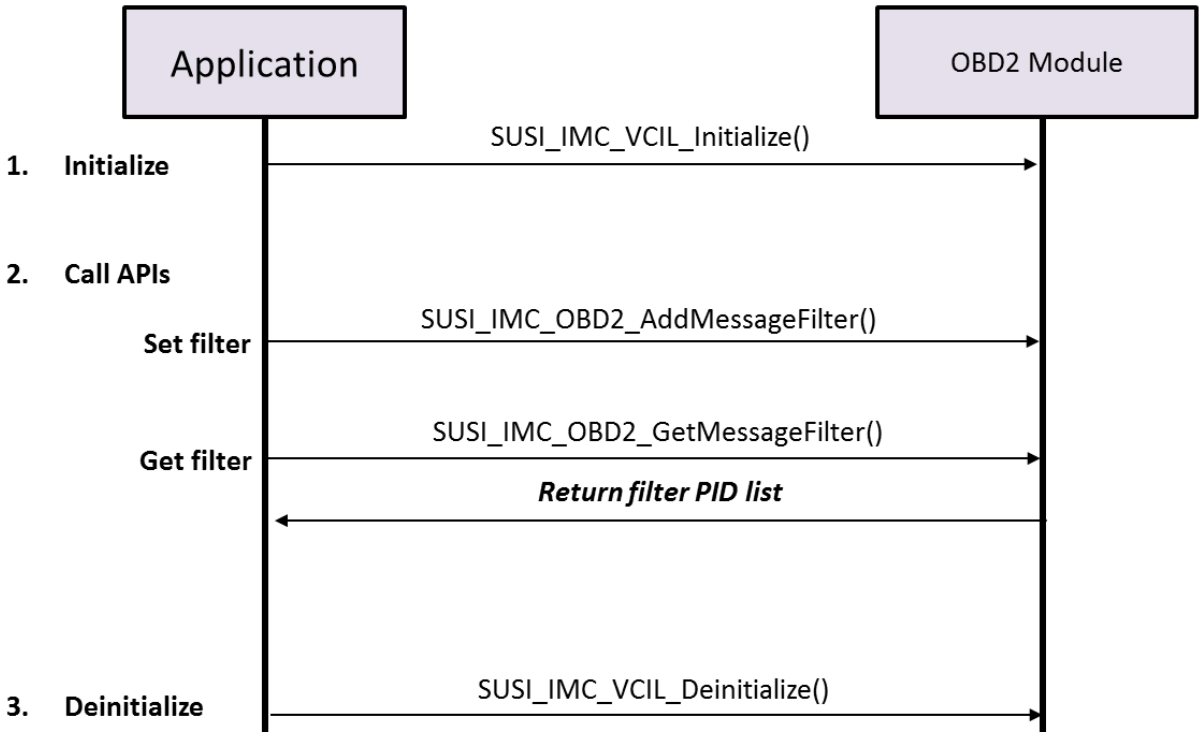
Remark:

The second byte **IMC_OBD2_MSG_OBJECT.**buf is PID.

Windows/ Linux/ Android

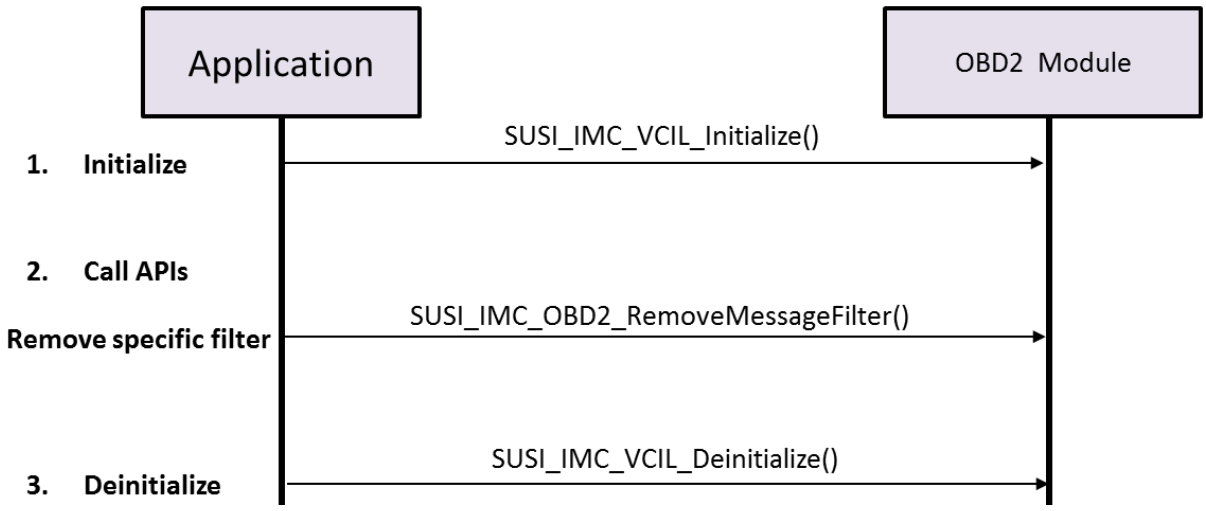
- **To get/set filter:**

Call SUSI_IMC_OBD2_AddMessageFilter() to set the mask and call SUSI_IMC_OBD2_GetMessageFilter() to get mask.



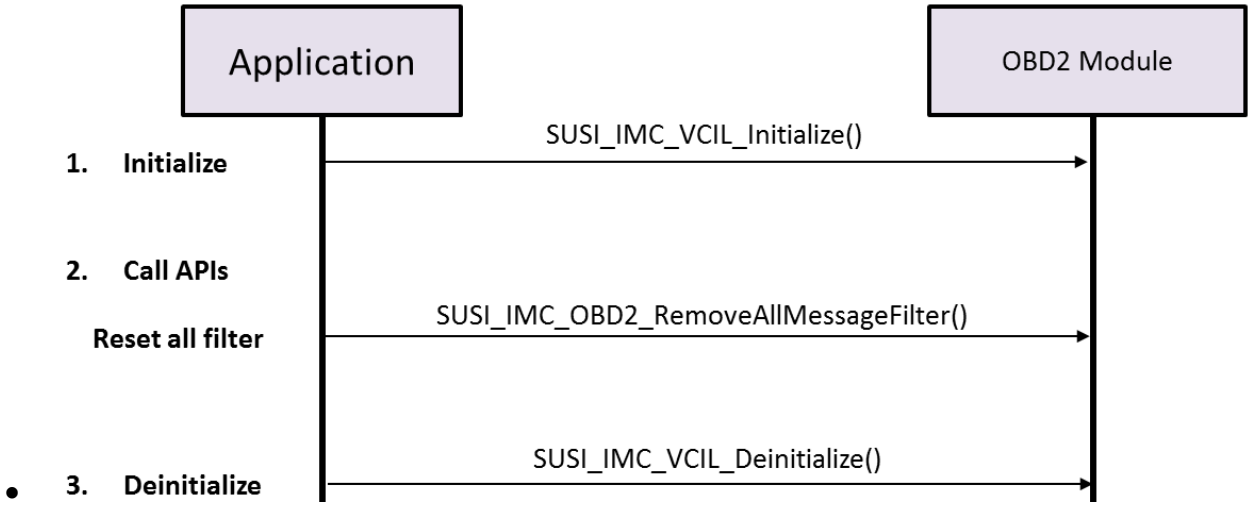
- **To remove specific filter:**

Call SUSI_IMC_OBD2_RemoveAllMessageFilter() to remove filter of specified filter ID.



- **To remove all filter:**

Call SUSI_IMC_OBD2_RemoveAllMessageFilter() to reset all filter.



•

5.14.2 Constant

5.14.2.1 Android

Class:
mrm.define.**IMC_CONSTANT**

Fields:

Field Name	Type	Value
MAX_OBD2_MESSAGE_BUFFER_SIZE	int	64
MAX_OBD2_FILTER_NUMBER	int	256

5.14.3 Structure/Class

5.14.3.1 Windows/Linux

IMC_OBD2_MSG_OBJECT Structure(deprecated)

Syntax:

```
typedef struct
{
        BYTE channel_number;
        BYTE dst;
        BYTE src;
        BYTE pri;
        BYTE tat;
        BYTE buf[64];
        BYTE buf_len;
}IMC_OBD2_MSG_OBJECT, *PIMC_OBD2_MSG_OBJECT;
```

Description:

This structure defines the OBD2 message.

Members:

channel_number

Physical channel number.

dst

Destination of message.

src

Source of message.

pri

Priority of message.

tat

Target address type

buf

Point to a receive buffer.

Buf_len

Data length code of the message (range from 0 to 64).

Remark:

The buf second byte is PID.

IMC_OBD2_MSG_EX_OBJECT Structure

Syntax:

```
typedef struct
{
        BYTE channel_number;
        BYTE type;
        DWORD id;
        USHORT buf_len;
        BYTE buf[MAX_OBD2_MESSAGE_BUFFER_SIZE];
}IMC_OBD2_MSG_EX_OBJECT, *PIMC_OBD2_MSG_EX_OBJECT;
```

Description:

This structure defines the OBD2 message.

Members:

channel_number

Physical channel number.

type

Type of message. 0 is 11-bits. 1 is 29-bits.

id

ISO 15765-4 OBD header in CAN identifier. The detail please refer to ISO 15765-4

buf_len

Data length code of the message (range from 0 to 64).

buf

Point to a buffer.

Remark:

The ISO 15765-4 (CAN) protocol in OBD2 Message format like below structure

<11-bits or 29-bits><PCI><7 data bytes>

The 11 bits ISO 15765-4 CAN standard define both functional addressing (ID/header = 0x7DF) and physical addressing (0x7En) for making request.

The 29 bits ISO 15765-4 CAN extend define as below

<Priority + reserved bits ><TAT><destination address><source address>

The priority with reserved byte normal is 0x18(Priority 6) . Since it is rarely changed. The next bytes describes the type of message that this is, and set to 0xDB for functional addressing, and 0xDA for physical addressing. The next two bytes is receiver and transmitter. For the functional diagnostic request, the receiver is always 0x33, and the transmitter is 0xF1.

5.14.3.2 Android

IMC_OBD2_MSG_OBJECT

class:

mrm.define.obd2.**IMC_OBD2_MSG_OBJECT**

Fields:

Field Name	Type	Input/Output	Comment
channel_number	byte	in	CAN channel number. The range is 1 to 2.
dst	byte	in, out	destination of message.
src	byte	in, out	source of message
pri	byte	in	priority of message.
tat	byte	in	target address type
buf	byte[]	in, out	Data of message. This field must be a byte array of length of MAX_OBD2_MESSAGE_BUFFER_SIZE(64)
buf_len	int	in, out	Data length of message. When writing message, the value indicates the bytes to be written from "buf". The value must not greater then length of "buf".

5.14.4 APIs

5.14.4.1 SUSI_IMC_OBD2_Read(deprecated)

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_OBD2_Read**(OUT PIMC_OBD2_MSG_OBJECT object);
Android	short **SUSI_IMC_OBD2_Read** (IMC_OBD2_MSG_OBJECT obd2MsgObject)

Description:

Read an 29-bits OBD2 messages from the system queue.

Parameters:

object [out]

Linux / Windows

Points to a variable that specifies the message of the OBD2 to be received.

Android

A instance of class **IMC_OBD2_MSG_OBJECT** for storing the returned received message

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

The default library message queue can store 2048 OBD2 message. If application not continue query the message from queue. The queue may overflow when the OBD2 bus data incoming continues. The OBD2 message may overwrite the last message when the queue with Insufficient space.

5.14.4.2 SUSI_IMC_OBD2_ReadEx

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_OBD2_ReadEx**(OUT PIMC_OBD2_MSG_EX_OBJECT object);
Android	short **SUSI_IMC_OBD2_ReadEx** (IMC_OBD2_MSG_EX_OBJECT obd2MsgObject)

Description:

Read an OBD2 messages from the system queue.

Parameters:

object [out]

Linux / Windows

Points to a variable that specifies the message of the OBD2 to be received.

Android

A instance of class **IMC_OBD2_MSG_EX_OBJECT** for storing the returned received message

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

firmware version must new than 2.14

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

The default library message queue can store 2048 OBD2 message. If application not continue query the message from queue. The queue may overflow when the OBD2 bus data incoming continues. The OBD2 message may overwrite the last message when the queue with Insufficient space.

5.14.4.3 SUSI_IMC_OBD2_Write((deprecated)

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_OBD2_Write**(IN PIMC_OBD2_MSG_OBJECT object);
Android	short **SUSI_IMC_OBD2_Write** (IMC_OBD2_MSG_OBJECT obd2MsgObject)

Description:

Write a message of OBD2 messages to the vehicle network.

Parameters:

object [in]

Linux / Windows

Points to a variable that specifies the message of the OBD2 to be sent.

Android

A instance of class **IMC_OBD2_MSG_OBJECT** for storing the message to be sent

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.14.4.4 SUSI_IMC_OBD2_WriteEx

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_OBD2_WriteEx**(IN PIMC_OBD2_MSG_EX_OBJECT object);
Android	short **SUSI_IMC_OBD2_WriteEx** (IMC_OBD2_MSG_EX_OBJECT obd2MsgObject)

Description:

Write a message of OBD2 messages to the vehicle network.

Parameters:

object [in]

Linux / Windows

Points to a variable that specifies the message of the OBD2 to be sent.

Android

A instance of class **IMC_OBD2_MSG_OBJECT** for storing the message to be sent

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

firmware version must new than 2.14

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.14.4.5 SUSI_IMC_OBD2_SetEvent

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_OBD2_SetEvent**(void* hEvent);

Description:

Set event to listen OBD2 data from bus. When VCIL receive a OBD2 message from bus, the event will be active. Event object is a HANDLE on Windows and the event object is pthread_cond_t on Linux.

Parameters:

hEvent [in]

Event handler of OBD2 message.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.14.4.6 SUSI_IMC_OBD2_SetEventHandler

Syntax:

Android	short **SUSI_IMC_OBD2_SetEventHandler**(Handler handler)

Description:

Set handler which handles OBD2 message received event.

Parameters:

handler [in]

An instance of Handler. Can not be null.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Remark:

- Please refer to the usage guide and sample code for for details.
- On alarm event triggered, the Handler will receive message with the "what" field equals to VCIL_EVENT_ID_RECEIVED_MSG_OBD2

5.14.4.7 SUSI_IMC_OBD2_UnsetEventHandler

Syntax:

Android	short **SUSI_IMC_OBD2_UnsetEventHandler**()

Description:

Unregister handler of OBD2 message received event.

Parameters:

none

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Remark:

- Please refer to the usage guide and sample code for for details.

5.14.4.8 SUSI_IMC_OBD2_WaitEvent

Syntax:

Android	short **SUSI_IMC_OBD2_WaitEvent**(boolean status)

Description:

Allow/Disallow VCIL to pass OBD2 message received event to registered handler when a message is pushed into SDK internal buffer.

Parameters:

status [in]

The status of whether VCIL should pass OBD2 message received event to registered handler

TRUE: Inform

FLASE: Not to inform

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Remark:

- Please refer to the usage guide and sample for for details.

5.14.4.9 SUSI_IMC_OBD2_AddMessageFilter

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_OBD2_AddMessageFilter**(IN BYTE channel_number, UINT pid);
Android	short **SUSI_IMC_OBD2_AddMessageFilter**(byte channelNumber, int pid)

Description:

Add the message pid to acceptance filter.

Parameters:

channel_number [in]

Linux / Windows / Android

Physical channel number. The range is 1 to 2.

pid [in]

Linux / Windows / Android

PID of filter to be added.

The range of pid is 0~255

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.14.4.10 SUSI_IMC_OBD2_GetMessageFilter

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_OBD2_GetMessageFilter**(OUT UINT channel_number, UINT* total, UINT* pid);
Android	short **SUSI_IMC_OBD2_GetMessageFilter**(byte channelNumber, int[] total, int[] pid)

Description:

Get massage pid filters.

Parameters:

channel_number [in]

Linux / Windows / Android

Physical channel number. The range is 1 to 2.

total [out]

Linux / Windows

Points to a variable that specifies the total number of PID filter.

Android

An allocated array of size 1 for storing the returned total number of PID filters.

The return value will be stored at index 0.

pid [out]

Linux / Windows

Points to a variable that specifies the message filters of the PID.

Android

An allocated array of size MAX_OBD2_FILTER_NUMBER (256) for storing the returned list of PID filters.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

Foer Windows/Linux, this API must to call 2 times. First, user must to assign the null to mid. It will return total number of mids. Second, user must to assign the array of pids with total number. After second step, this API will return all message filters.

5.14.4.11 SUSI_IMC_OBD2_RemoveMessageFilter

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_OBD2_RemoveMessageFilter**(IN BYTE channel_number, IN UINT pid);
Android	short **SUSI_IMC_OBD2_RemoveMessageFilter**(byte channelNumber, int pid)

Description:

Remove message filter.

Parameters:

channel_number [in]

Linux / Windows / Android

Physical channel number. The range is 1 to 2.

pid [in]

Linux / Windows / Android

PID of filter to be removed

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.14.4.12 SUSI_IMC_OBD2_RemoveAllMessageFilter

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_OBD2_RemoveAllMessageFilter** (IN BYTE channel_number);
Android	short **SUSI_IMC_OBD2_RemoveAllMessageFilter**(byte channelNumber)

Description:

Reset all specified physical channel filters to the default value.

Parameters:

channel_number [in]

Linux / Windows / Android

Physical channel number. The range is 1 to 2.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.15 J1708

5.15.1 Usage Overview

TREK device supports data transmission of J1708 protocol and provides filter functions.
The SDK hold a internal buffer for received J1708 messages with limited size. You can use the read API to consume the buffer and get received message. When the internal buffer become full before application consume it, the oldest message will be removed and new message will be added.

The following section describe the usage of corresponding APIs.

5.15.1.1 Basic Usage

Windows/Linux

To start using J1708 APIs you should first initialize the APIs and activate J1708 protocol on specific J1708 channel.

Please refer to the VCIL usage for how to initialize and configure J1708 channels.

Android

The class for J1708 APIs for Android is **mrm.VCIL**
To start using J1708 APIs you should first initialize the APIs and activate J1708 protocol on specific J1708 channel .

Please refer to the VCIL usage for basic usage and for how to initialize and configure J1708 channels.

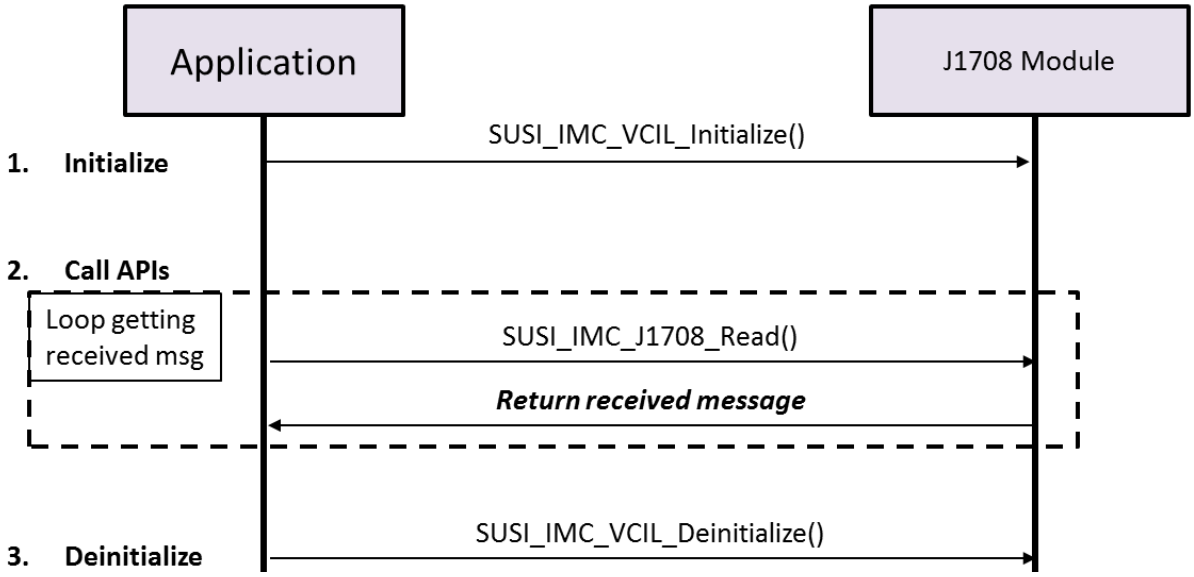
5.15.1.2 J1708 Message Reading

The mechanism of reading J1708 messages from J1708 bus by using SDK APIs is basically the same with CAN. Please refer to the section CAN Message Reading . This section will only shows the brief flow of using J1708 APIs.

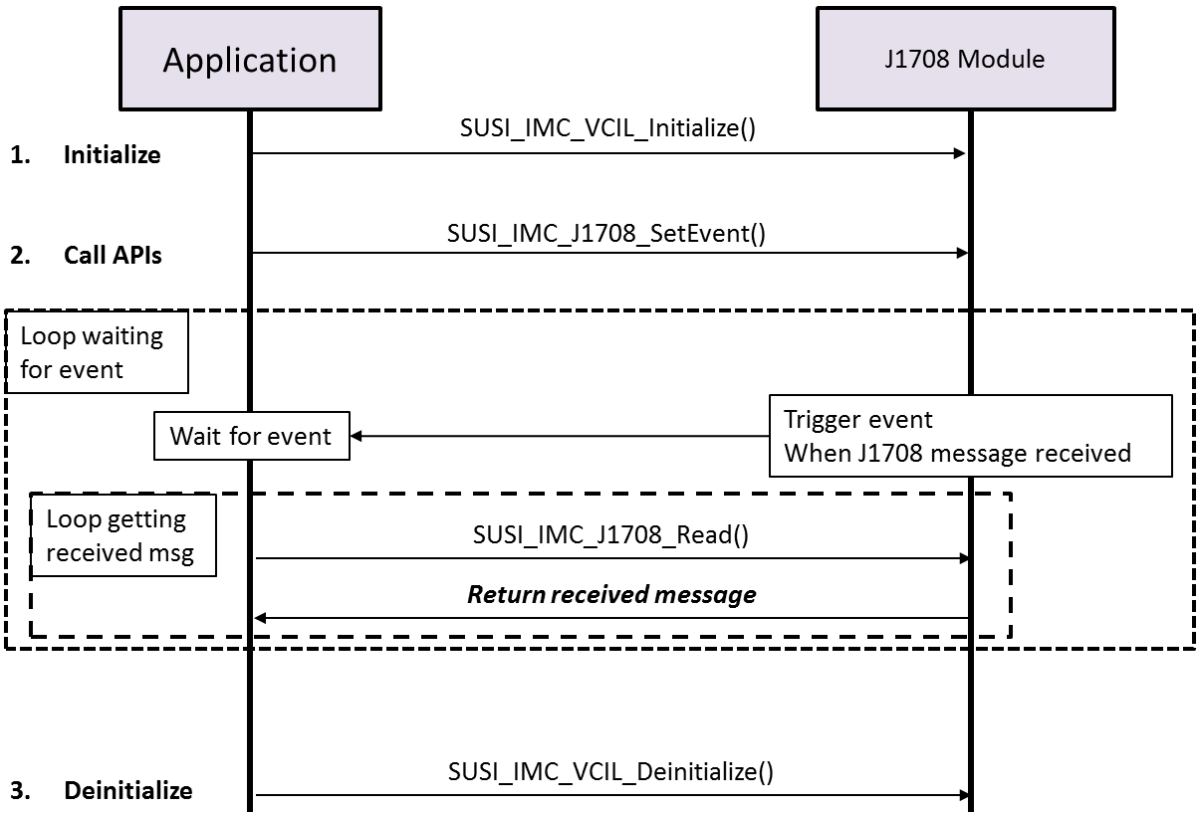
Windows/Linux

You implement your J1708 message reading application in either **Polling** or **Event Handling** style.

- **Polling:**



- **Event Handling:**

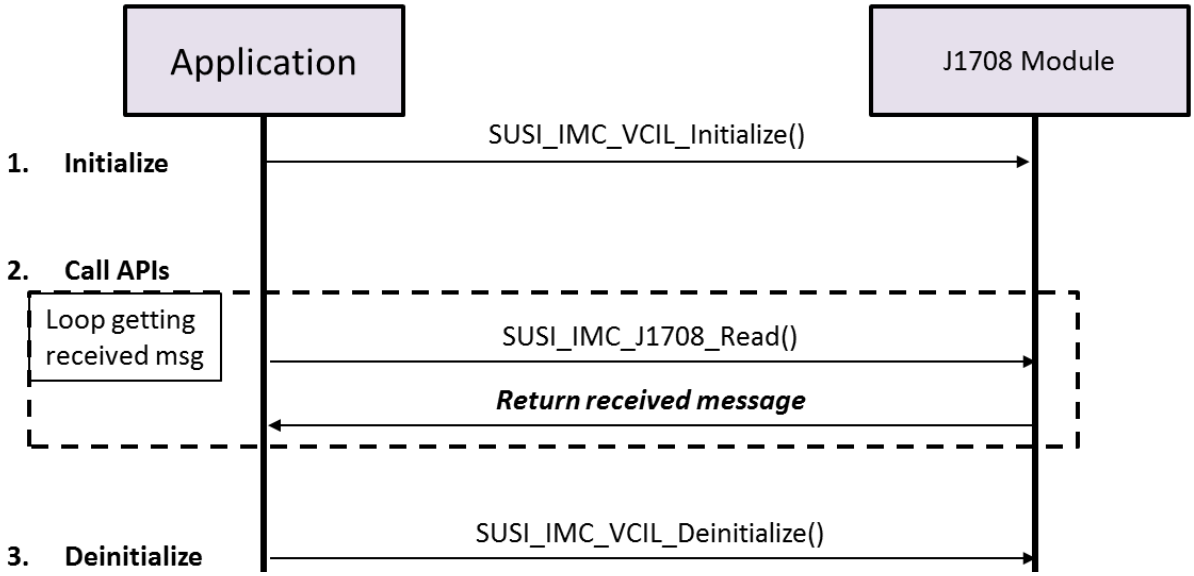

Application
J1708 Module
1. Initialize
SUSI_IMC_VCIL_Initialize()
2. Call APIs
SUSI_IMC_J1708_SetEvent()
Loop waiting for event
Wait for event
Trigger event
When J1708 message received
Loop getting received msg
SUSI_IMC_J1708_Read()
Return received message
3. Deinitialize
SUSI_IMC_VCIL_Deinitialize()

Android

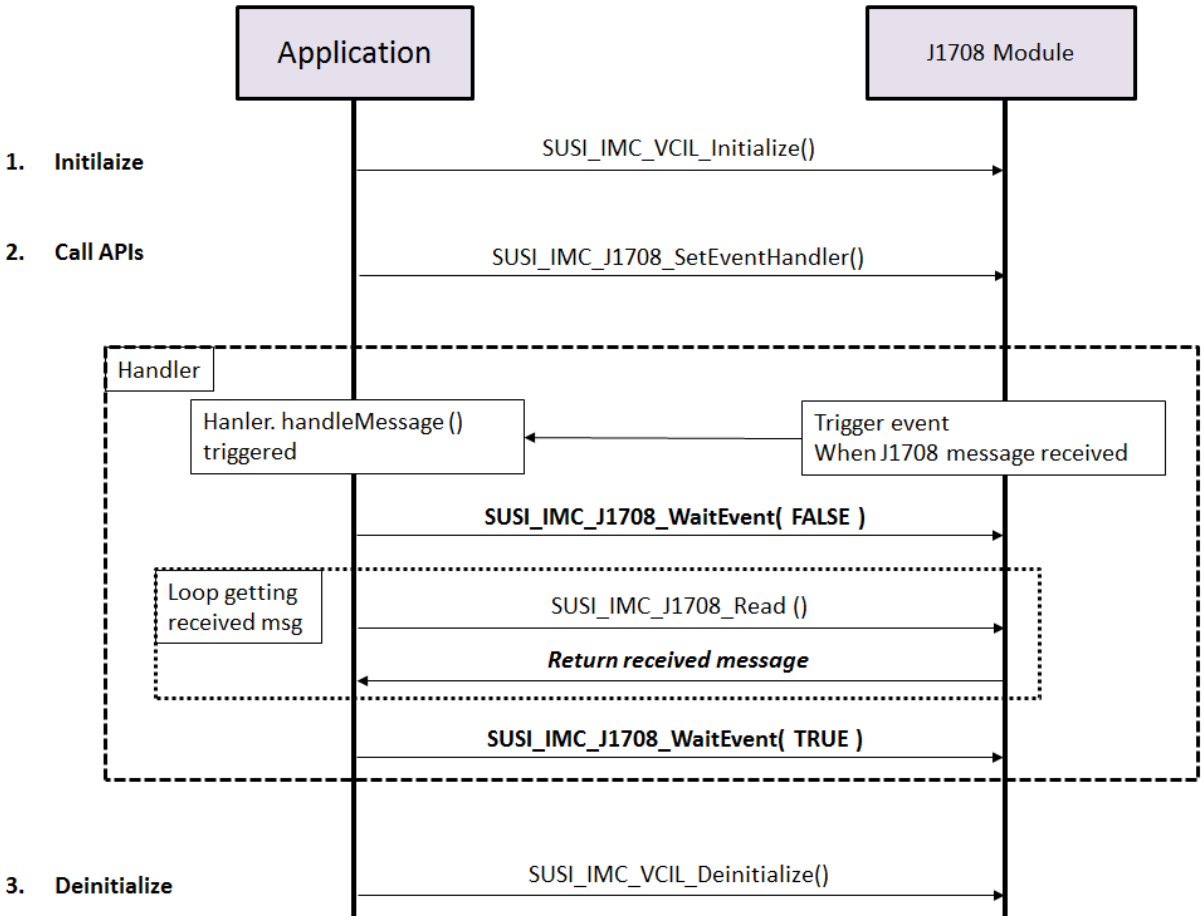
For the details of usage, please also refer to CAN Message Reading for Android.

You implement your J1708 message reading application in either **Polling** or **Event Handling** style.

- **Polling:**



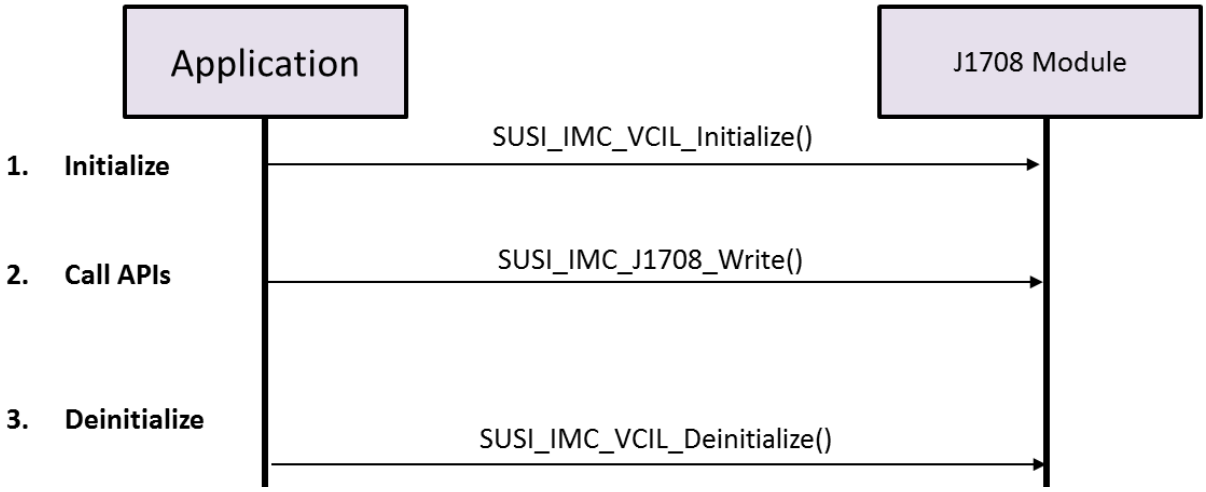
- **Event Handling:**


Application
J1708 Module
1. Initilaize
SUSI_IMC_VCIL_Initialize()
2. Call APIs
SUSI_IMC_J1708_SetEventHandler()
Handler
Hanler. handleMessage () triggered
Trigger event When J1708 message received
SUSI_IMC_J1708_WaitEvent(FALSE)
Loop getting received msg
SUSI_IMC_J1708_Read ()
Return received message
SUSI_IMC_J1708_WaitEvent(TRUE)
3. Deinitialize
SUSI_IMC_VCIL_Deinitialize()

5.15.1.3 J1708 Message Writing

This section describe how to write J1708 messages to J1708 bus by using SDK APIs.

Windows/ Linux/ Android



To write J1708 message to J1708 bus, you can call SUSI_IMC_J1708_Write() with a IMC_J1708_MSG_OBJECT structure which contains the data you want to write.

5.15.1.4 J1708 Acceptance filter Settings

Similar with CAN Acceptance Filter, the VCIL MCU provide acceptance filter for J1708 protocol, which can filter the MID of J1708 message. The VCIL MCU supports at most 256 MID filters. A J1708 message with MID which is matched with any of the filters will be accepted by MCU.

To enable the filter function, you **must configure the J1708 physical channel to VCIL_MODE_J1708_WITH_MASK, to work with filter first**. Please refer to the Basic Usage of VCIL section.

The following is an example of filter setting. If we set filter with MID 0x80, 0x56, 0x11 When the received J1708 message's MID value is 0x56, the messages is accepted.

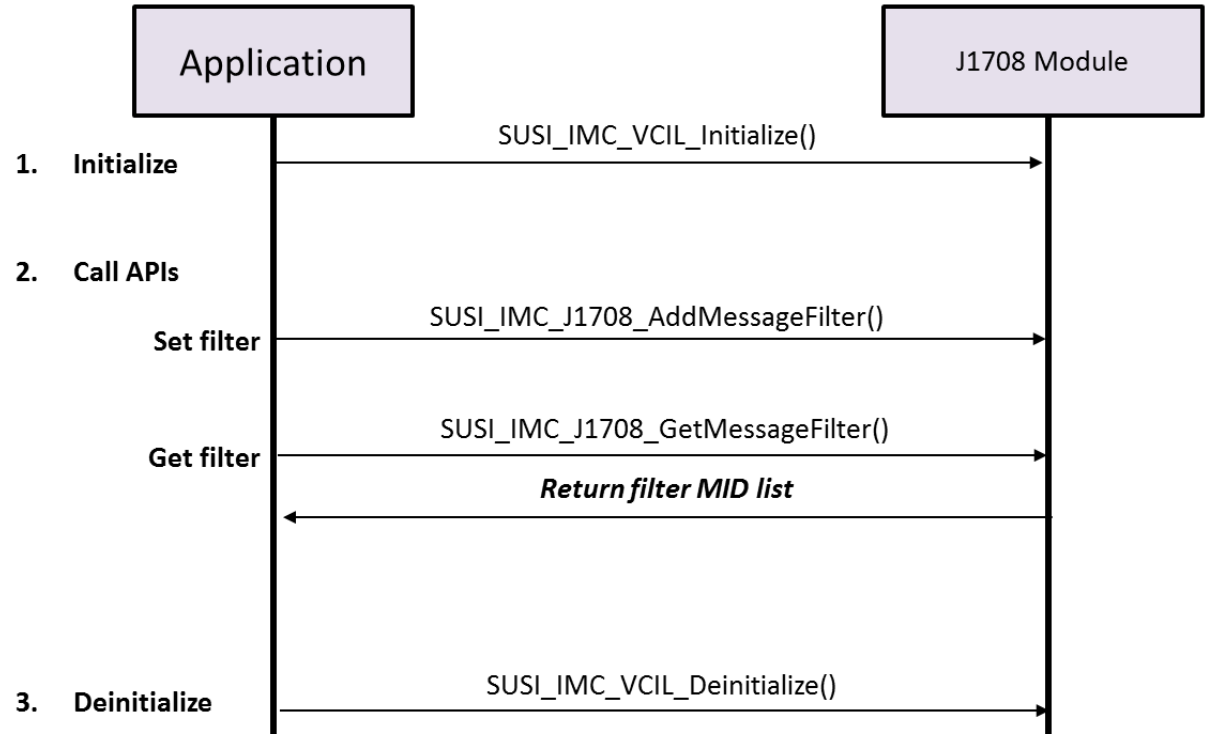
J1708 Filter Configuration	Receive J1708 data's MID value	Result
Filter MID=0x80	0x80	**Accept**
Filter MID=0x80	0x79	**Drop**

J1708 Filter Configuration	Receive J1708 data's MID value	Result
Filter MID=0x80 OR 0x56 OR 0x11	0x80	**Accept**
Filter MID=0x80 OR 0x56 OR 0x11	0x56	**Accept**
Filter MID=0x80 OR 0x56 OR 0x11	0x57	**Drop**
Filter MID=0x80 OR 0x56 OR 0x11	0x12	**Drop**

Windows/ Linux/ Android

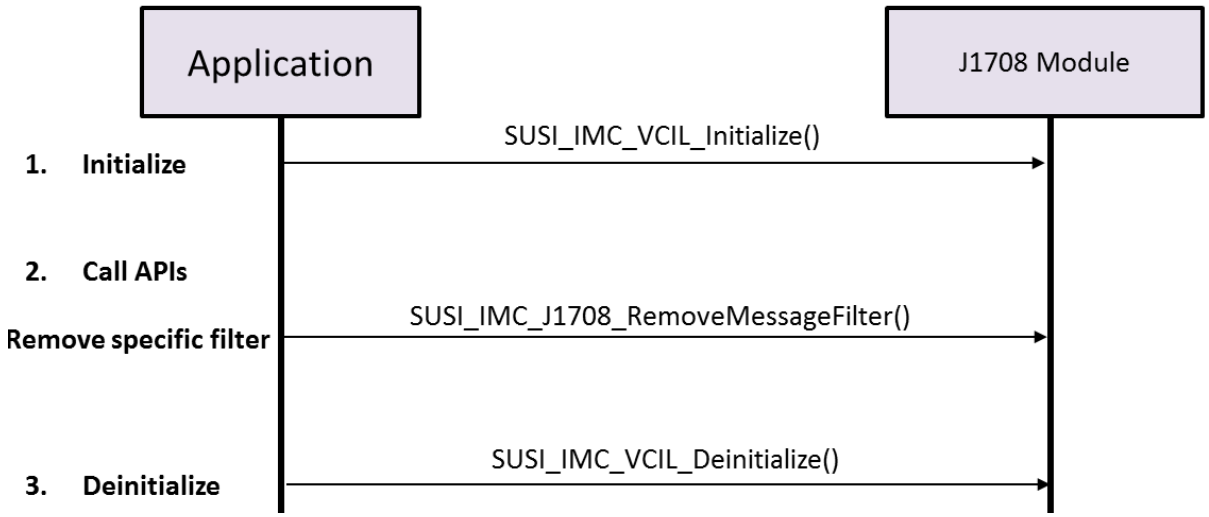
- **To get/set filter:**

Call SUSI_IMC_J1708_AddMessageFilter() to set the mask and call SUSI_IMC_J1708_GetMessageFilter() to get mask.



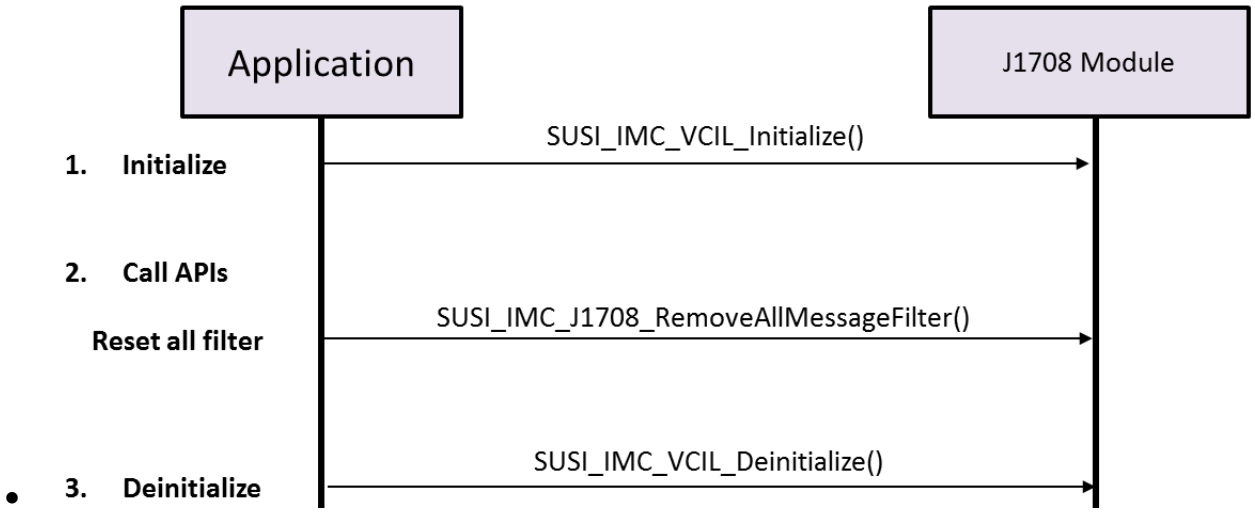
- **To remove specific filter:**

Call SUSI_IMC_J1708_RemoveMessageFilter() to remove filter of specified filter ID.



- **To remove all filter:**

Call SUSI_IMC_J1708_RemoveAllMessageFilter() to reset all filter.


Application
J1708 Module
1. Initialize
SUSI_IMC_VCIL_Initialize()
2. Call APIs
Reset all filter
SUSI_IMC_J1708_RemoveAllMessageFilter()
3. Deinitialize
SUSI_IMC_VCIL_Deinitialize()

•

5.15.2 Constant

5.15.2.1 Android

Class:
mrm.define.**IMC_CONSTANT**

Fields:

Field Name	Type	Value
MAX_J1708_MESSAGE_BUFFER_SIZE	int	20
MAX_J1708_FILTER_NUMBER	int	256

5.15.3 Structure/Class

5.15.3.1 Windows/Linux

IMC_J1708_MSG_OBJECT Structure

Syntax:

```
typedef struct
{
        UINT mid;
            UINT pid;
            BYTE pri;
            BYTE buf[20];
            BYTE buf_len;
}IMC_J1708_MSG_OBJECT, *PIMC_J1708_MSG_OBJECT;
```

Description:

This data structure defines the J1708 message.

Members:

mid

The message identification character. The range is 0x00 to 0xFF.

pid

The parameter Identification.

pri

The priority of this message. The priority of range is 1 to 8. Normally set to 6.

buf

Point to a receive buffer.

Buf_len

Data length code of the message (range from 0 to 20).

5.15.3.2 Android

IMC_J1708_MSG_OBJECT

class:

mrm.define.j1708.**IMC_J1708_MSG_OBJECT**

Fields:

Field Name	Type	Input/Output	Comment
mid	int	in	The message identification character. The range is 0x00 to 0xFF.
pid	int	in, out	The parameter Identification.
pri	byte	in	The priority of this message. The priority of range is 1 to 8. Normally set to 6.
buf	byte[]	in, out	Data of message. This field must be a byte array of length of MAX_J1708_MESSAGE_BUFFER_SIZE(20)
buf_len	int	in, out	Data length of message. When writing message, the value indicate the bytes to be written from "buf". The value must not greater then length of "buf" and should follow **SAE-J1708.**

5.15.4 APIs

5.15.4.1 SUSI_IMC_J1708_Read

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_J1708_Read**(OUT PIMC_J1708_MSG_OBJECT object);
Android	short **SUSI_IMC_J1708_Read** (IMC_J1708_MSG_OBJECT j1708MsgObject)

Description:

Read a J1708 messages from the library queue.

Parameters:

object [out]

Linux / Windows

Points to a variable that specifies the message of the J1708 to be received.

Android

A instance of class **IMC_J1708_MSG_OBJECT** for storing the returned received message

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

The default library message queue can store 2048 J1708 message. If application not continue query the message from queue. The queue may overflow when the J1708 bus data incoming continues. The J1708 message may overwrite the last message when the queue with Insufficient space.

5.15.4.2 SUSI_IMC_J1708_Write

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_J1708_Write**(IN PIMC_J1708_MSG_OBJECT object);
Android	short **SUSI_IMC_J1708_Write** (IMC_J1708_MSG_OBJECT j1708MsgObject)

Description:

Write a message of J1708 messages to the vehicle network.

Parameters:

object [in]

Linux / Windows

Points to a variable that specifies the message of the J1708 to be sent.

Android

A instance of class **IMC_J1708_MSG_OBJECT** for storing the message to be sent

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.15.4.3 SUSI_IMC_J1708_SetEvent

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_J1708_SetEvent**(void* hEvent);

Description:

Set event to listen J1708 data from bus. When VCIL receive a J1708 message from bus, the event will be active. Event object is a HANDLE on Windows and the event object is pthread_cond_t on Linux.

Parameters:

hEvent [in]

Event handler of J1708 message.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.15.4.4 SUSI_IMC_J1708_SetEventHandler

Syntax:

Android	short **SUSI_IMC_J1708_SetEventHandler**(Handler handler)

Description:

Set handler which handles J1708 message received event.

Parameters:

handler [in]

An instance of Handler. Can not be null.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Remark:

- Please refer to the usage guide and sample code for for details.
- On alarm event triggered, the Handler will receive message with the "what" field equals to VCIL_EVENT_ID_RECEIVED_MSG_J1708

5.15.4.5 SUSI_IMC_J1708_UnsetEventHandler

Syntax:

Android	short **SUSI_IMC_J1708_UnsetEventHandler**()

Description:

Unregister handler of J1708 message received event.

Parameters:

none

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Remark:

- Please refer to the usage guide and sample code for for details.

5.15.4.6 SUSI_IMC_J1708_WaitEvent

Syntax:

Android	short **SUSI_IMC_J1708_WaitEvent**(boolean status)

Description:

Allow/Disallow VCIL to pass J1708 message received event to registered handler when a message is pushed into SDK internal buffer.

Parameters:

status [in]

The status of whether VCIL should pass J1708 message received event to registered handler

TRUE: Inform

FLASE: Not to inform

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Remark:

- Please refer to the usage guide and sample for for details.

5.15.4.7 SUSI_IMC_J1708_AddMessageFilter

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_J1708_AddMessageFilter**(IN UINT mid);
Android	short **SUSI_IMC_J1708_AddMessageFilter**(int mid)

Description:

Add the message mid to acceptance filter.

Parameters:

UINT mid [in]

Linux / Windows / Android

Points to a variable that specifies the message filter of MID to be added.

The range of mid is 0~255

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.15.4.8 SUSI_IMC_J1708_GetMessageFilter

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_J1708_GetMessageFilter**(OUT UINT* total, OUT UINT* mid);
Android	short **SUSI_IMC_J1708_GetMessageFilter**(int[] total, int[] mid)

Description:

Get massage mid filters.

Parameters:

total [out]

Linux / Windows

Points to a variable that specifies the total number of MID filter.

Android

An allocated array of size 1 for storing the returned total number of MID filters.

The return value will be stored at index 0.

pid [out]

Linux / Windows

Points to a variable that specifies the message filters of the MID.

Android

An allocated array of size MAX_J1708_FILTER_NUMBER (256) for storing the returned list of MID filters.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

This API must to call 2 times. First, user must to assign the null to mid. It will return total number of mids. Second, user must to assign the array of mids with total number. After second step, this API will return all message filters.

5.15.4.9 SUSI_IMC_J1708_RemoveMessageFilter

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_J1708_RemoveMessageFilter**(IN UINT mid);
Android	short **SUSI_IMC_J1708_RemoveMessageFilter**(int mid)

Description:

Remove message filter.

Parameters:

mid [in]

Linux / Windows / Android

MID of filter to be removed

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.15.4.10 SUSI_IMC_J1708_RemoveAllMessageFilter

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_J1708_RemoveAllMessageFilter** (void);
Android	short **SUSI_IMC_J1708_RemoveAllMessageFilter**()

Description:

Reset all message filters to default.

Parameters:

None

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.16 J1587

5.16.1 Usage Overview

TREK device supports data transmission of J1587 protocol and provides filter functions.
The SDK hold a internal buffer for received J1587 messages with limited size. You can use the read API to consume the buffer and get received message. When the internal buffer become full before application consume it, the oldest message will be removed and new message will be added.

The following section describe the usage of corresponding APIs.

5.16.1.1 Basic Usage

Windows/Linux

To start using J1587 APIs you should first initialize the APIs and activate J1587 protocol on specific J1587 channel.
Please refer to the VCIL usage for how to initialize and configure J1587 channels.

Android

The class for J1587 APIs for Android is **mrm.VCIL**
To start using J1587 APIs you should first initialize the APIs and activate J1587 protocol on specific J1587 channel .
Please refer to the VCIL usage for basic usage and for how to initialize and configure J1587 channels.

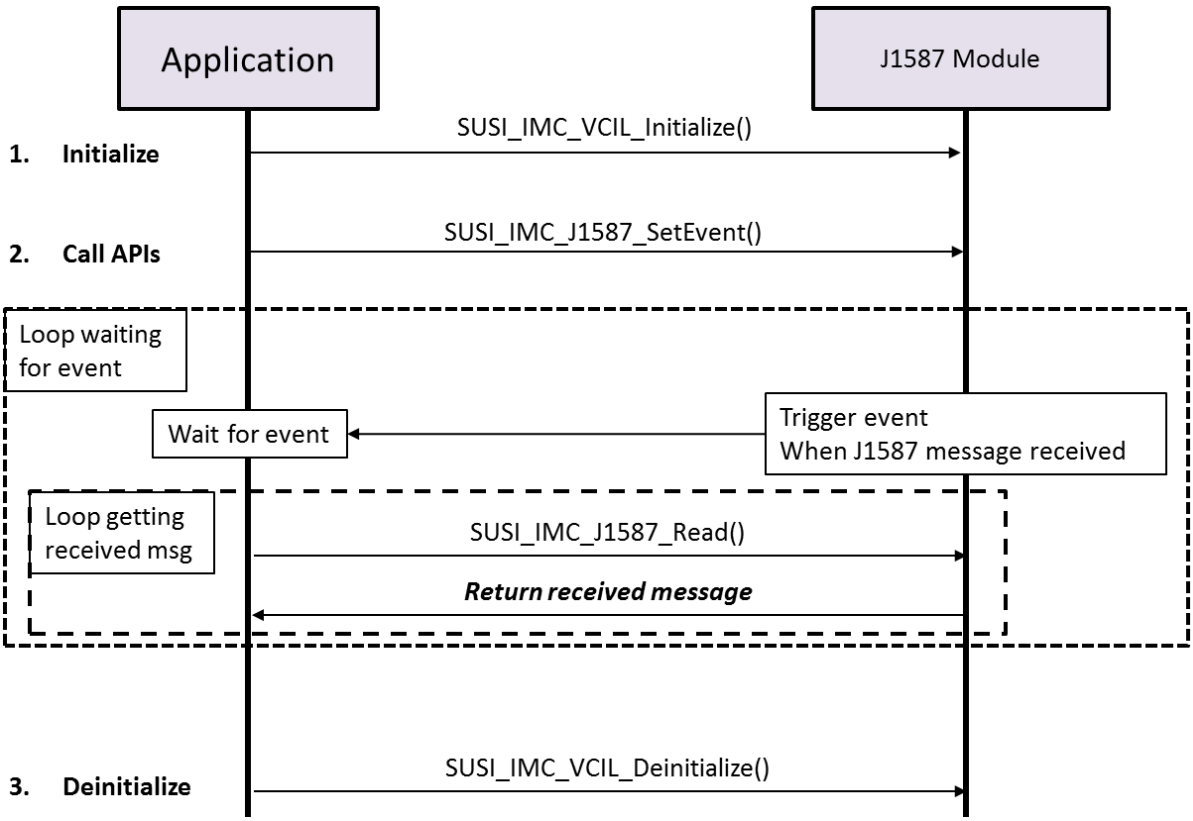
5.16.1.2 J1587 Message Reading

The mechanism of reading J1587 messages from J1708 bus by using SDK APIs is basically the same with CAN. Please refer to the section CAN Message Reading . This section will only shows the brief flow of using J1708 APIs.

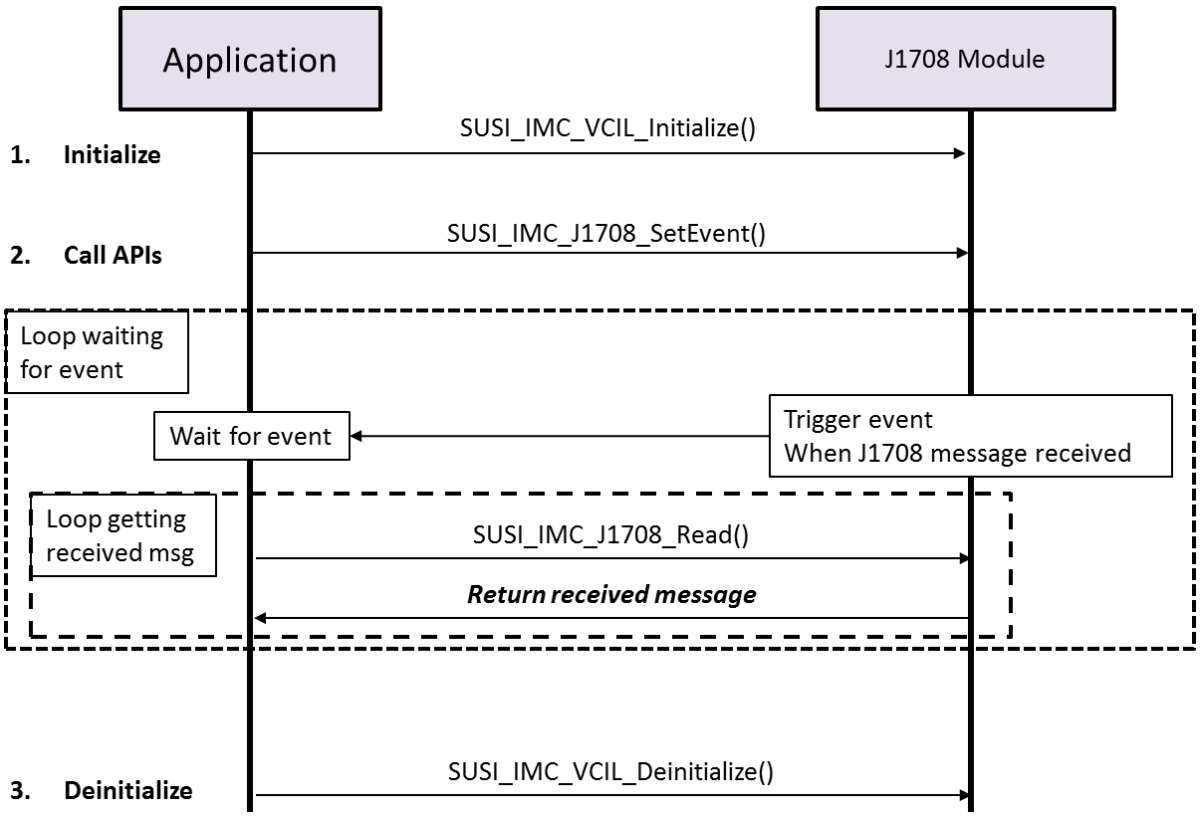
Windows/Linux

You implement your J1708 message reading application in either **Polling** or **Event Handling** style.

- **Polling:**



- **Event Handling:**

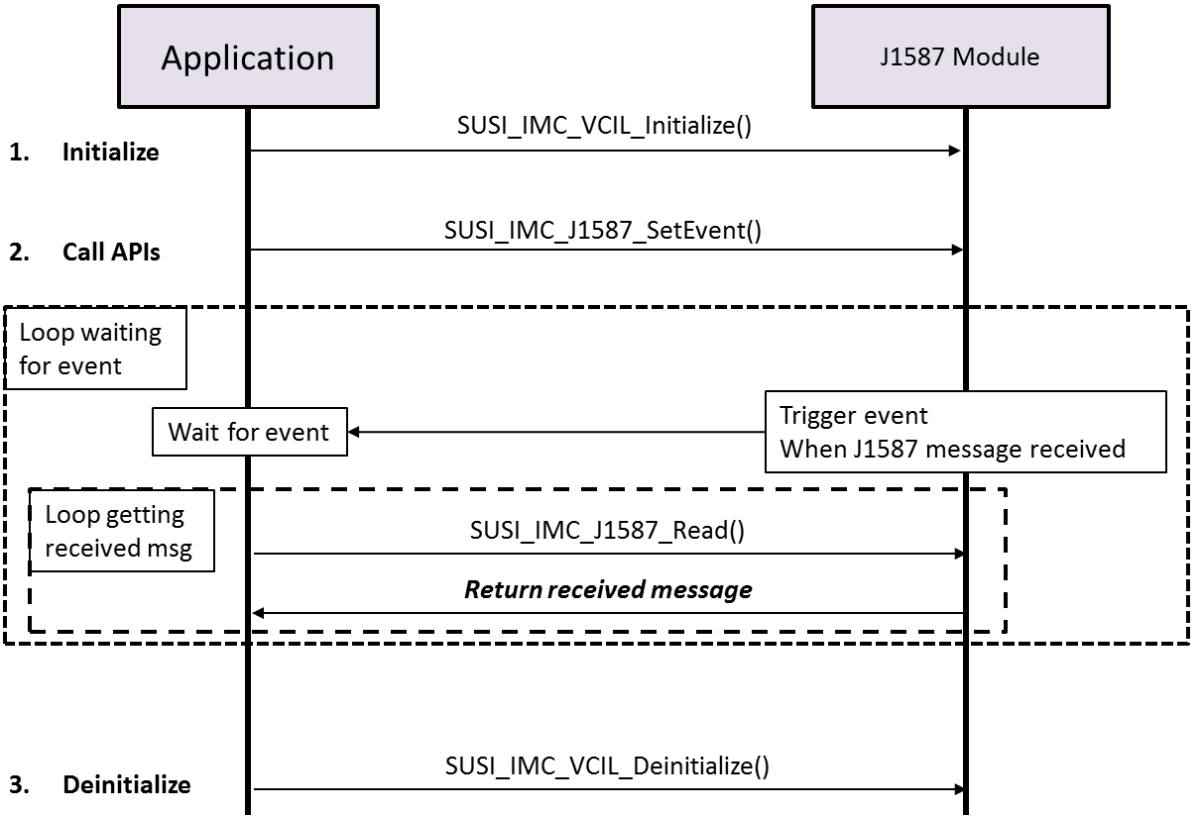

Application
J1708 Module
1. Initialize
SUSI_IMC_VCIL_Initialize()
2. Call APIs
SUSI_IMC_J1708_SetEvent()
Loop waiting for event
Wait for event
Trigger event
When J1708 message received
Loop getting received msg
SUSI_IMC_J1708_Read()
Return received message
3. Deinitialize
SUSI_IMC_VCIL_Deinitialize()

Android

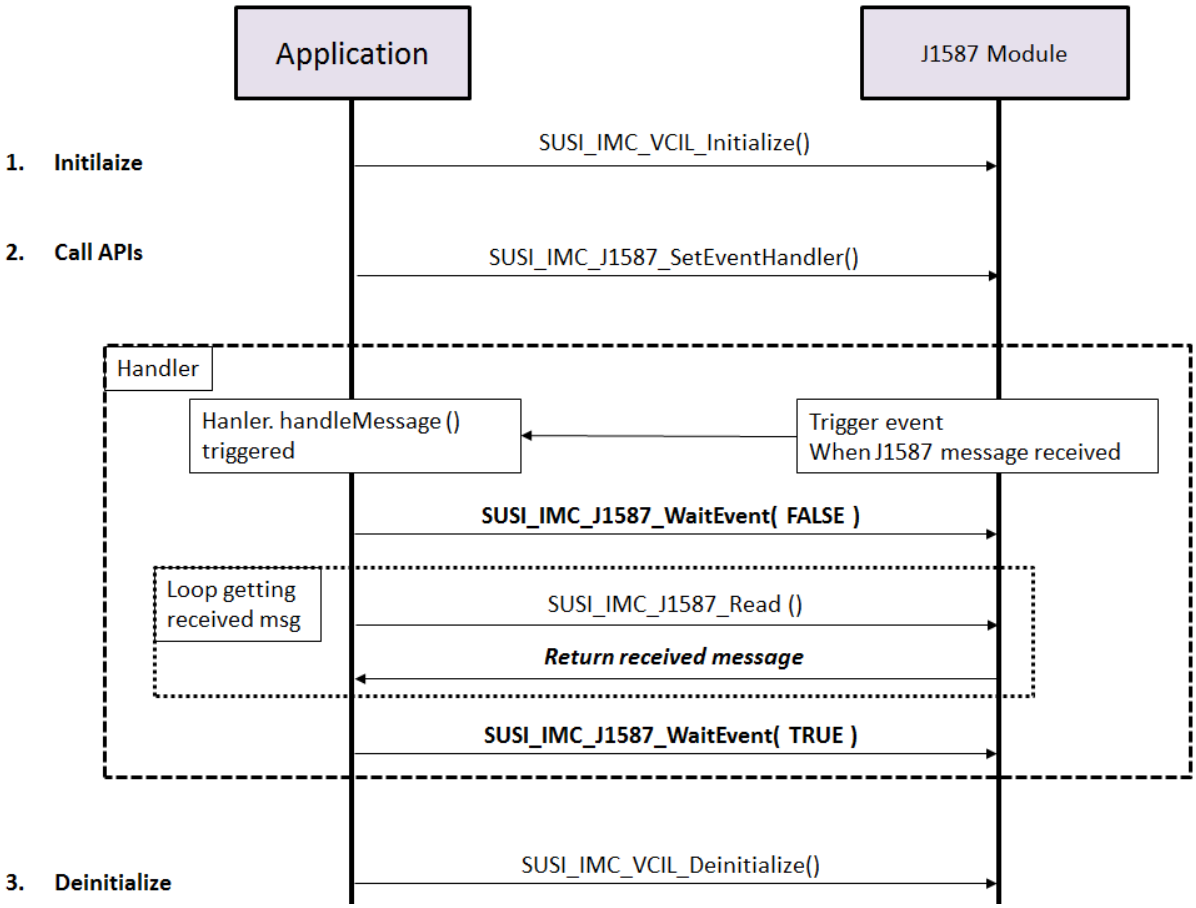
For the details of usage, please also refer to CAN Message Reading for Android.

You implement your J1708 message reading application in either **Polling** or **Event Handling** style.

- **Polling:**



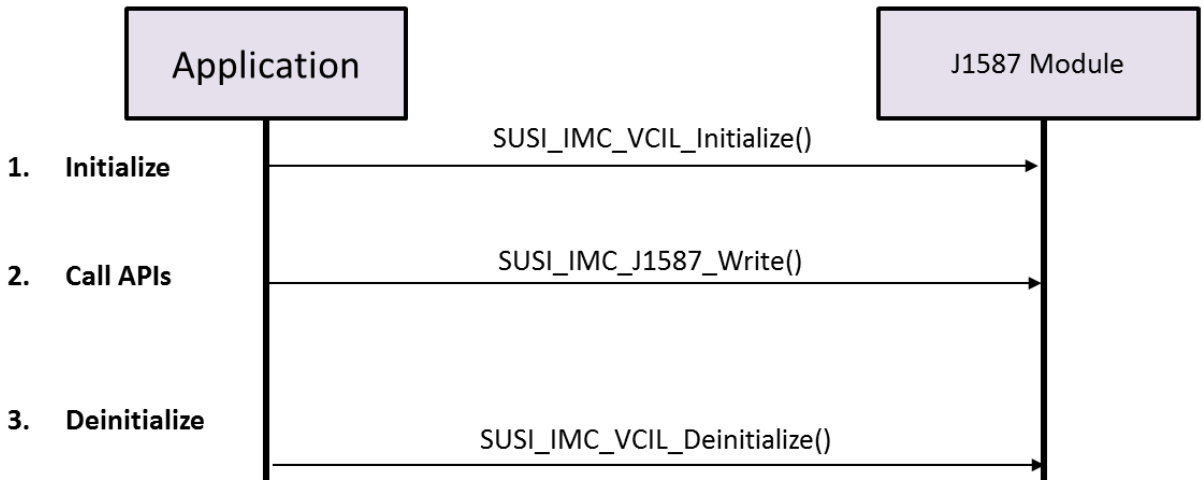
- **Event Handling:**


Application
J1587 Module
1. Initilaize
SUSI_IMC_VCIL_Initialize()
2. Call APIs
SUSI_IMC_J1587_SetEventHandler()
Handler
Hanler. handleMessage () triggered
Trigger event When J1587 message received
SUSI_IMC_J1587_WaitEvent(FALSE)
Loop getting received msg
SUSI_IMC_J1587_Read ()
Return received message
SUSI_IMC_J1587_WaitEvent(TRUE)
3. Deinitialize
SUSI_IMC_VCIL_Deinitialize()

5.16.1.3 J1587 Message Writing

This section describe how to write J1708 messages to J1708 bus by using SDK APIs.

Windows/ Linux/ Android



To write J1587 message to J1708 bus, you can call SUSI_IMC_J1587_Write() with a IMC_J1587_MSG_OBJECT structure which contains the data you want to write.

5.16.1.4 J1587 Acceptance filter Settings

Similar with CAN Acceptance Filter, the VCIL MCU provide acceptance filter for J1587 protocol, which can filter the PID of J1587 message. The VCIL MCU supports at most 512 PID filters. A J1587 message with PID which is matched with any of the filters will be accepted by MCU.

To enable the filter function, you **must configure the J1708 physical channel to VCIL_MODE_J1587_WITH_MASK, to work with filter first**. Please refer to the Basic Usage of VCIL section.

The following is an example of filter setting. If we set filter with PID 0x80, 0x56, 0x11 When the received J1587 message's PID value is 0x56, the messages is accepted.

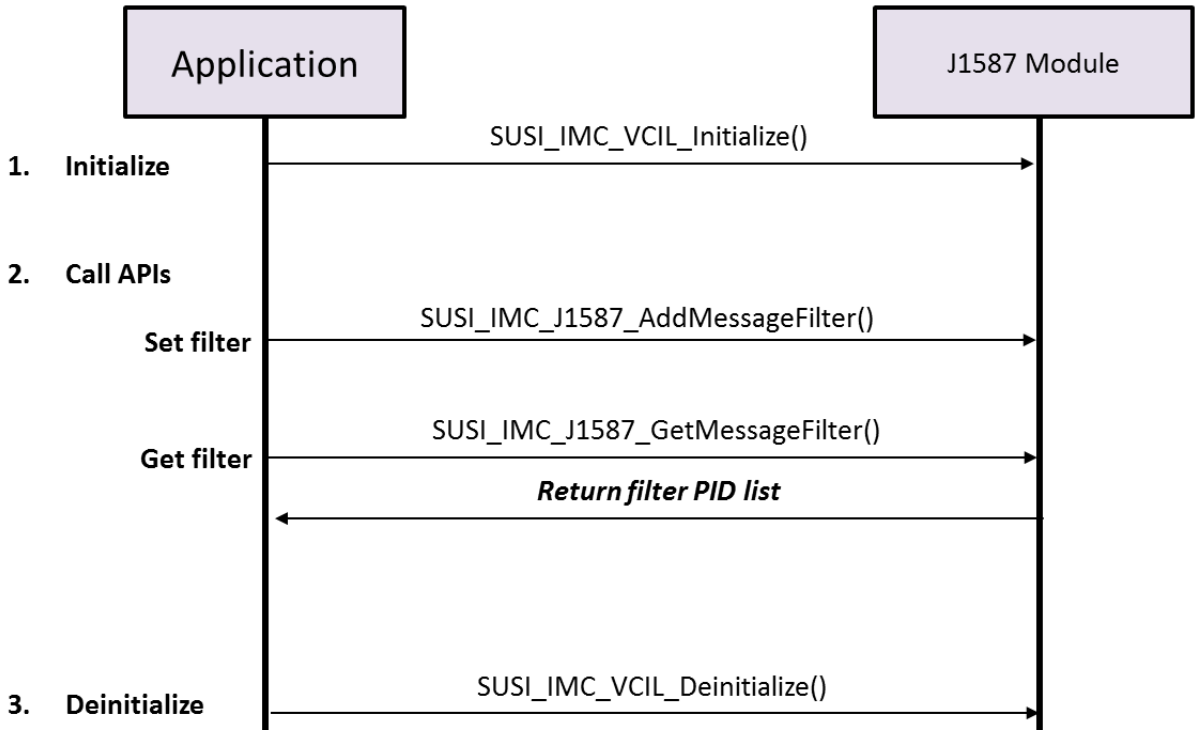
J1587 Filter Configuration	Receive J1587 data's PID value	Result
Filter PID=0x80	0x80	**Accept**
Filter PID=0x80	0x79	**Drop**

J1587 Filter Configuration	Receive J1587 data's PID value	Result
Filter PID=0x80 OR 0x56 OR 0x11	0x80	**Accept**
Filter PID=0x80 OR 0x56 OR 0x11	0x56	**Accept**
Filter PID=0x80 OR 0x56 OR 0x11	0x57	**Drop**
Filter PID=0x80 OR 0x56 OR 0x11	0x12	**Drop**

Windows/ Linux/ Android

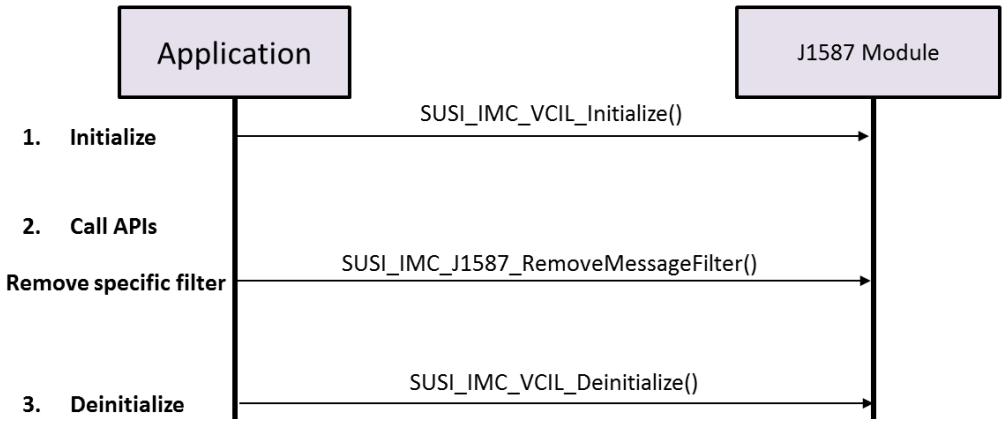
- **To get/set filter:**

Call SUSI_IMC_J1587_AddMessageFilter() to set the mask and call SUSI_IMC_J1587_GetMessageFilter() to get mask.



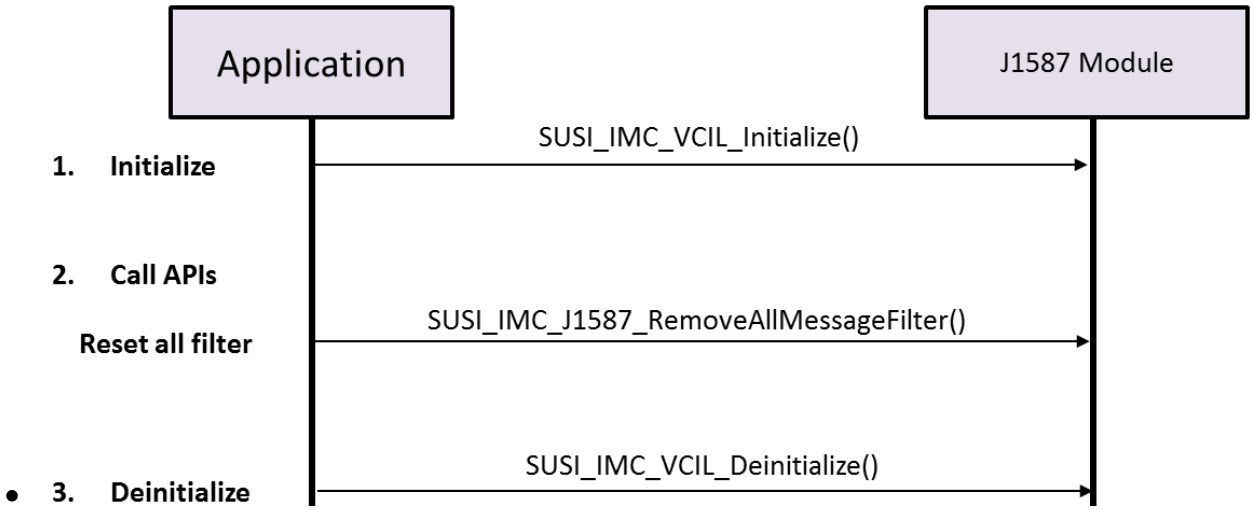
- **To remove specific filter:**

Call SUSI_IMC_J1587_RemoveMessageFilter() to remove filter of specified filter ID.



- **To remove all filter:**

Call SUSI_IMC_J1587_RemoveAllMessageFilter() to reset all filter.


Application
J1587 Module
1. Initialize
SUSI_IMC_VCIL_Initialize()
2. Call APIs
Reset all filter
SUSI_IMC_J1587_RemoveAllMessageFilter()
3. Deinitialize
SUSI_IMC_VCIL_Deinitialize()

5.16.2 Constant

5.16.2.1 Android

Class:
mrm.define.**IMC_CONSTANT**

Fields:

Field Name	Type	Value
MAX_J1587_MESSAGE_BUFFER_SIZE	int	20
MAX_J1587_FILTER_NUMBER	int	512

5.16.3 Structure/Class

5.16.3.1 Windows/Linux

IMC_J1587_MSG_OBJECT Structure

Syntax:

```
typedef struct
{
        UINT mid;
            UINT pid;
            BYTE pri;
            BYTE buf[20];
            BYTE buf_len;
}IMC_J1587_MSG_OBJECT, *PIMC_J1587_MSG_OBJECT;
```

Description:

This data structure defines the J1587 message.

Members:

mid

The message identification character. The range is 0x00 to 0xFF.

pid

The parameter Identification. The range is 0x00 to 0x1FE, exclude 0xFF. The detail range please refer to **SAE-J1587**

pri

The priority of this message. The priority of range is 1 to 8. Normally set to 6.

buf

Point to a receive buffer.

Buf_len

Data length code of the message (range from 0 to 20).

5.16.3.2 Android

IMC_J1587_MSG_OBJECT

class:

mrm.define.j1587.**IMC_J1587_MSG_OBJECT**

Fields:

Field Name	Type	Input/Output	Comment
mid	int	in	The message identification character. The range is 0x00 to 0xFF.
pid	int	in, out	The parameter Identification. The range is 0x00 to 0x1FE, exclude 0xFF. The detail range please refer to **SAE-J1587**
pri	byte	in	The priority of this message. The priority of range is 1 to 8. Normally set to 6.
buf	byte[]	in, out	Data of message. This field must be a byte array of length of MAX_J1587_MESSAGE_BUFFER_SIZE(20)
buf_len	int	in, out	Data length of message. When writing message, the value indicate the bytes to be written from "buf". The value must not greater then length of "buf" and should follow **SAE-J1587.**

5.16.4 APIs

5.16.4.1 SUSI_IMC_J1587_Read

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_J1587_Read**(OUT PIMC_J1587_MSG_OBJECT object);
Android	short **SUSI_IMC_J1587_Read** (IMC_J1587_MSG_OBJECT j1587MsgObject)

Description:

Read a J1587 messages from the library queue.

Parameters:

object [out]

Points to a variable that specifies the message of the J1587 to be received.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

The default library message queue can store 2048 J1587 message. If application not continue query the message from queue. The queue may overflow when the J1587 bus data incoming continues. The J1587 message may overwrite the last message when the queue with Insufficient space.

5.16.4.2 SUSI_IMC_J1587_Write

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_J1587_Write**(IN PIMC_J1587_MSG_OBJECT object);
Android	short **SUSI_IMC_J1587_Write** (IMC_J1587_MSG_OBJECT j1587MsgObject)

Description:

Write a message of J1587 messages to the vehicle network.

Parameters:

object [out]

Points to a variable that specifies the message of the J1587 to be received.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.16.4.3 SUSI_IMC_J1587_SetEvent

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_J1587_SetEvent**(void* hEvent);

Description:

Set event to listen J1587 data from data bus. When VCIL receive a J1587 message from bus, the event will be active. Event object is a HANDLE on Windows and the event object is pthread_cond_t on Linux.

Parameters:

hEvent [in]

Event handler of J1587 message.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.16.4.4 SUSI_IMC_J1587_SetEventHandler

Syntax:

Android	short **SUSI_IMC_J1708_SetEventHandler**(Handler handler)

Description:

Set handler which handles J1587 message received event. Can not be null.

Parameters:

handler [in]

An instance of Handler.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Remark:

- Please refer to the usage guide and sample code for for details.
- On alarm event triggered, the Handler will receive message with the "what" field equals to VCIL_EVENT_ID_RECEIVED_MSG_J1587

5.16.4.5 SUSI_IMC_J1587_UnsetEventHandler

Syntax:

Android	short **SUSI_IMC_J1587_UnsetEventHandler**()

Description:

Unregister handler of J1587 message received event.

Parameters:

none

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Remark:

- Please refer to the usage guide and sample code for for details.

5.16.4.6 SUSI_IMC_J1587_WaitEvent

Syntax:

Android	short **SUSI_IMC_J1587_WaitEvent**(boolean status)

Description:

Allow/Disallow VCIL to pass J1587 message received event to registered handler when a message is pushed into SDK internal buffer.

Parameters:

status [in]

The status of whether VCIL should pass J1587 message received event to registered handler

TRUE: Inform

FLASE: Not to inform

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Remark:

- Please refer to the usage guide and sample for for details.

5.16.4.7 SUSI_IMC_J1587_AddMessageFilter

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_J1587_AddMessageFilter**(IN UINT pid);
Android	short **SUSI_IMC_J1587_AddMessageFilter**(int mid)

Description:

Add the message PID to acceptance filter.

Parameters:

UINT pid [in]

Points to a variable that specifies the message filter of pid to be added.

The range of pid is 0~510 exclude 255.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

PID=255 & PID=511 use for page extend defined by SAE. You should not use it direct.

5.16.4.8 SUSI_IMC_J1587_GetMessageFilter

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_J1587_GetMessageFilter**(OUT UINT* total, OUT UINT* pid);
Android	short **SUSI_IMC_J1587_GetMessageFilter**(int[] total, int[] mid)

Description:

Get massage PID filters.

Parameters:

UINT* total [out]

Points to a variable that specifies the total number of message filter of the pid.

UINT* pid [out]

Points to a variable that specifies the message filters of the pid.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

For Windows/Linux, this API must to call 2 times. First, user msut to assign the null to pid. It will return total number of PID. Second, user must to assign the array of pids with total number. After second step, this API will return all message filters.

5.16.4.9 SUSI_IMC_J1587_RemoveMessageFilter

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_J1587_RemoveMessageFilter**(IN UINT pid);
Android	short **SUSI_IMC_J1587_RemoveMessageFilter**(int mid)

Description:

Remove message PID filter.

Parameters:

pid [in]

PID of J1587 message.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

PID=255 & PID=511 use for page extend defined by SAE. You should not use it direct.

5.16.4.10 SUSI_IMC_J1587_RemoveAllMessageFilter

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_J1587_RemoveAllMessageFilter** (void);
Android	short **SUSI_IMC_J1587_RemoveAllMessageFilter**()

Description:

Reset all message filters to default.

Parameters:

None

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.17 SDP SRV

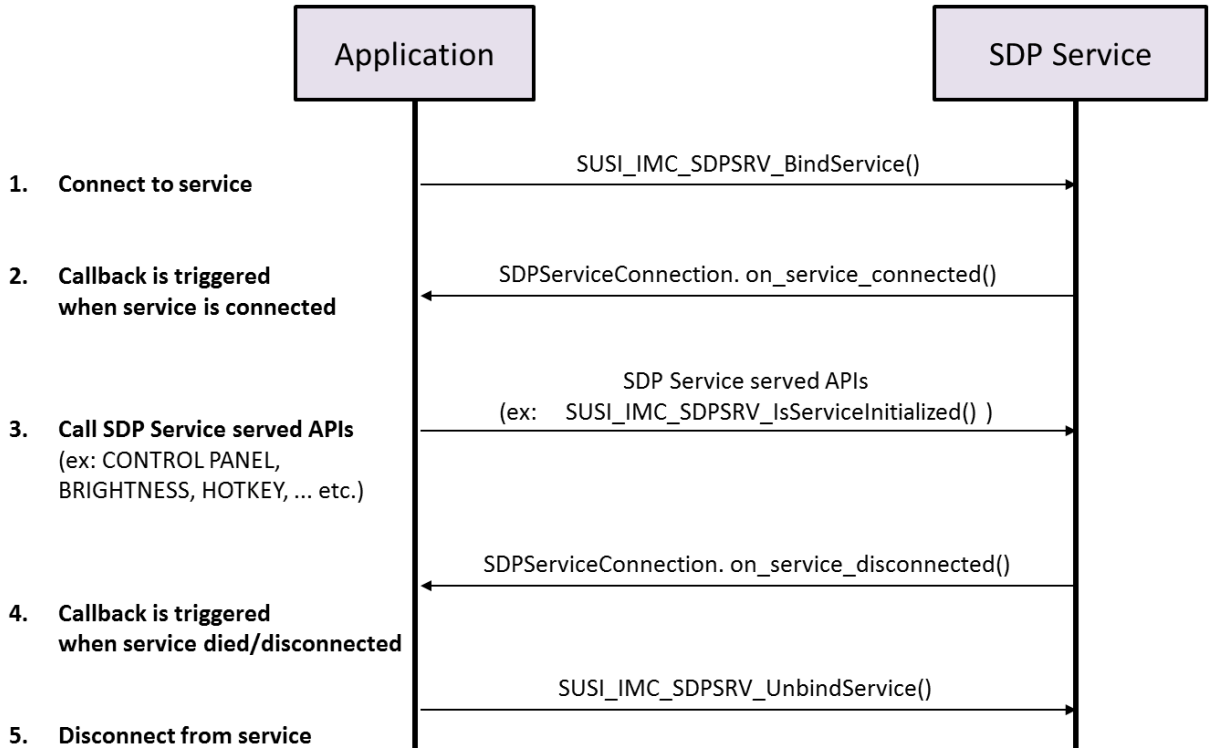
5.17.1 Usage Overview

In TREK SDK for Android, the SDP SRV, CONTROL PANEL, LIGHT SENSOR, BRIGHTNESS, HOTKEY modules are served by SDP Service. Before using these API, you must connect your APP process to the SDP Service.

The SDPSRV API module provides APIs to connect/disconnect APP to/from SDP service, and also APIs for getting status of SDP service.

The following section describes the usage of APIs.

5.17.1.1 Android



The class for SDPSRV APIs for Android is **mrm.client.SDPServiceClient**.

To use APIs served by SDP Service (SDP SRV, CONTROL PANEL, LIGHT SENSOR, HOTKEY, DISPLAY) , you must first call SUSI_IMC_SDPSRV_BindService() to connect your client application process to SDP Service.

As the argument of SUSI_IMC_SDPSRV_BindService(), you must create a instance of class - SDPServiceConnection and implement the methods:

- **on_service_connected**() and
- **on_service_disconnected**()

to define the behavior of your application's behavior when service is connected and disconnected.

After **SUSI_IMC_SDPSRV_BindService**() is called, the client application will try to connect to the SDP Service running in background. If SDP Service is not running, the service will be started and all API modules served by SDP Service will be initialized.

If client application process is connected to service correctly, the implemented **SDPServiceConnection.on_service_connected**() will be callback. After the callback, you can then start calling APIs served by SDP Service (ex: SDP SRV, CONTROL PANEL, LIGHT SENSOR, BRIGHTNESS, HOTKEY).

If SDP Service died or disconnected abnormally, the implemented **SDPServiceConnection.on_service_disconnected**() will be callback so your application process will be informed.

To disconnect your application from SDP service, you can call SUSI_IMC_SDPSRV_UnbindService() to release allocated resources in SDP Service. You should always disconnect your application from service before the applications is closed.

For details, please refer to the sample code - SDPSample.

5.17.2 Constant

5.17.2.1 Android

Class:
mrm.define.**IMC_CONSTANT**

Fields:

Field Name	Type	Value
SDP_EVENT_ID_UNKNOWN	int	-1
SDP_EVENT_ID_HOTKEY_PRESS_STATUS_CHANGED	int	0

5.17.3 Structure/Classes

5.17.3.1 Android

SDPServiceConnection

class:

mrm.client.**SDPServiceConnection**

Fields:

-

Methods:

Syntax	Comment
abstract void **on_service_connected**()	The callback which will be triggered when client application process is connected to SDP Service is successfully after SUSI_IMC_SDPSRV_BindService() is called. You must implement this method to define the your application's behavior when service connected.
abstract void **on_service_disconnected**()	The callback which will be triggered when SDP Service is successfully died or disconnected abnormally. You must implement this method to define the your application's behavior when service disconnected.

Remark:
For details of implementation, please refer to the sample code - SDPSample.

5.17.4 APIs

5.17.4.1 SUSI_IMC_SDPSRV_BindService

Syntax:

Android	short **SUSI_IMC_SDPSRV_BindService**(SDPServiceConnection conn)

Description:

Connect client application process to running SDP Service,
If SDP Service is not running when this API is called, the service will be started and initialized.

Parameters:

conn [in]

Instance of SDPServiceConnection.

You must implement

SDPServiceConnection.**on_service_connected**()/**on_service_disconnected**() to define the your application's behavior when service connected/disconnected.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Android

Remark:

-

5.17.4.2 SUSI_IMC_SDPSRV_UnbindService

Syntax:

Android	short **SUSI_IMC_SDPSRV_UnbindService**()

Description:

Disconnect client application process from SDP Service and release allocated resources.

Parameters:

-

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Android

Remark:

-

5.17.4.3 SUSI_IMC_SDPSRV_IsServiceInitialized

Syntax:

Android	short **SUSI_IMC_SDPSRV_IsServiceInitialized**(boolean[] status)

Description:

Get initialization status of SDP Service.

Parameters:

status

An allocated array of size 1 for storing returned initialization status.

The returned value will be store at index 0.

The value is TRUE, if SDP Service is initialized and all served APIs are available.

The value is FALSE, if SDP Service is not initialized.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Android

Remark:

-

5.17.4.4 SUSI_IMC_SDPSRV_IsServiceConnected

Syntax:

Android	boolean **SUSI_IMC_SDPSRV_IsServiceConnected**()

Description:

Check whether the client application process is connected to SDP Service.

Parameters:

-

Returns:

Returns TRUE, if connected.

Returns FALSE, if disconnected.

Supported Platforms:

TREK-570 Android

Remark:

-

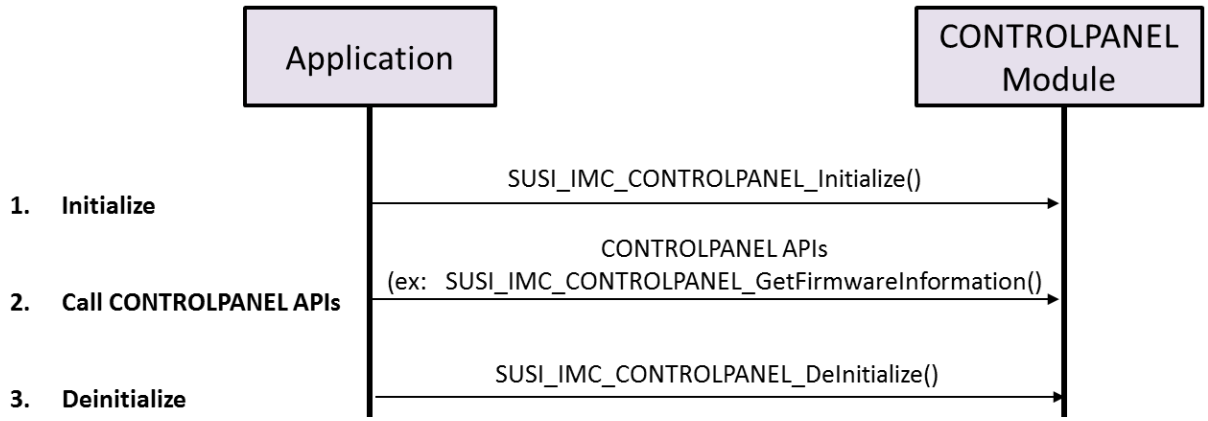
5.18 CONTROL PANEL

5.18.1 Usage Overview

The CONTROL PANEL, LIGHT SENSOR, BRIGHTNESS, HOTKEY API modules are used to control the external smart display device (ex: TREK-306, TREK-303). Where the CONTROL PANEL API module is the basic module which provides APIs for getting basic information of smart display device and some basic operations.

The following section describes the usage of APIs.

5.18.1.1 Windows/Linux



To use CONTROLPANEL APIs. You need call SUSI_IMC_CONTROLPANEL_Initialize() to Initialize the CONTROLPANEL module first. Then you can start using CONTROLPANEL APIs.

The CONTROL PANEL should be initialized first before you calling any APIs of the LIGHT SENSOR, BRIGHTNESS, HOTKEY API modules.

To stop using the CONTROLPANEL module must call SUSI_IMC_CONTROLPANEL_Deinitialize() to close CONTROLPANEL module .

5.18.1.2 Android

The class for CONTROLPANEL APIs for Android is **mrm.client.SDPServiceClient**.

The APP should be connected to SDP service before you calling any APIs of the CONTROL PANEL, LIGHT SENSOR, BRIGHTNESS, HOTKEY API modules.

Please refer to the SDP SRV Usage Overview for basic usage.

5.18.2 Enumeration

5.18.2.1 Android

CONTROL_PANEL_RESET_MODE enum

class:

mrm.define.IMC_ENUM.**CONTROL_PANEL_RESET_MODE**

Enum:

Name	Type	value	Comment
SUSI_RESET_INTO_NORMAL_MODE	byte	0x00	
SUSI_RESET_INTO_BOOT_LOADER_MODE	byte	0x01	

Remark:

Use the method **getValue()** to get the enum value.

ex:

MRM_ENUM.CONTROL_PANEL_RESET_MODE.SUSI_RESET_INTO_BOOT_LOADER_MODE**.getValue()** returns 1.

Please refer to sample code for detailed usage.

5.18.3 Structure/Classes

5.18.3.1 Windows/Linux

IMC_CONTROLPANEL_FIRMWARE_INFO Structure

Syntax:

```
typedef struct
{
        BYTE version[3];
            BYTE model_name[20];
}IMC_CONTROLPANEL_FIRMWARE_INFO,*PIMC_CONTROLPANEL_FIRMWARE_INFO;
```

Description:

This structure defines the firmware info of control panel .

Members:

version

Firmware version as below,

Byte1:Main version

Byte2:Sub version

Byte3:Revision version

model_name

Firmware model name length is fixed to 20 bytes.

If the model name is less than 20 bytes, the other bytes must fill in 0x00.

Default model name in this firmware is FW-303H

5.18.3.2 Android

IMC_CONTROLPANEL_FIRMWARE_INFO

class:

mrm.define.controlpanel.**IMC_CONTROLPANEL_FIRMWARE_INFO**

Fields:

Field Name	Type	Input/Output	Comment
version	byte[]	out	Firmware version. The element definitions are as below: Byte1:Main version Byte2:Sub version Byte3:Revision version
model_name	String	out	Model name.

5.18.4 APIs

5.18.4.1 SUSI_IMC_CONTROLPANEL_GetVersion

Syntax:

Windows	USHORT **SUSI_IMC_CONTROLPANEL_GetVersion** (OUT PBYTE version);

Description:

Get current version from the Control Panel library.

Parameters:

version [out]

Point to read buffer that specifies the Control Panel library's current version.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Windows

TREK-572 Windows

TREK-674 Windows

Remark:

The version buffer size must be greater than 17 bytes.

5.18.4.2 SUSI_IMC_CONTROLPANEL_Initialize

Syntax:

Linux / Windows	USHORT **SUSI_IMC_CONTROLPANEL_Initialize** (IN BYTE port_number);

Description:

Initialize the Control Panel library for Hotkey, Brightness, and Light Sensor.

Parameters:

port_number [in]

Specifies the serial port which communication with TREK-303

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

Before using any Hotkey, Brightness and Light Sensor API functions, must call this function first.

It shown the UART mapping table as following:

Hardware Platform	OS Platform	Port ID
TREK-674	Windows Linux	3
TREK-570	Windows Linux	1
TREK-572	Windows Linux	2

5.18.4.3 SUSI_IMC_CONTROLPANEL_Deinitialize

Syntax:

Linux / Windows	USHORT **SUSI_IMC_CONTROLPANEL_Deinitialize** (void);

Description:

Deinitialize the Control Panel library from system memory.

Parameters:

None

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

Notify: Call this function before unloading the Control Panel library from memory.

5.18.4.4 SUSI_IMC_CONTROLPANEL_GetFirmwareInformation

Syntax:

Linux / Windows	USHORT **SUSI_IMC_CONTROLPANEL_GetFirmwareInformation** (OUT PIMC_CONTROLPANEL_FIRMWARE_INFO info);
Android	short **SUSI_IMC_CONTROLPANEL_GetFirmwareInformation**(IMC_CONTROLPANEL _FIRMWARE_INFO info)

Description:

Get the firmware information from TREK-303.

Parameters:

info [out]

Windows:

Points to a variable that specifies the firmware information of the TREK-303.

Android:

Instance of IMC_CONTROLPANEL_FIRMWARE_INFO for storing returned value;

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.18.4.5 SUSI_IMC_CONTROLPANEL_ResetFirmware

Syntax:

Linux / Windows	USHORT **SUSI_IMC_CONTROLPANEL_ResetFirmware** (IN BYTE mode);
Android	short **SUSI_IMC_CONTROLPANEL_ResetFirmware**(byte mode)

Description:

Reset the TREK-303H/L firmware.

Parameters:

mode [in]

The working mode after reset.

Windows:

The available value is:

SUSI_RESET_INTO_NORMAL_MODE

SUSI_RESET_INTO_BOOT_LOADER_MODE

Android:

Available values are enumerated in class CONTROL_PANEL_RESET_MODE

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows

TREK-572 Linux / Windows / Android

TREK-674 Linux / Windows

Remark:

SUSI_RESET_INTO_BOOT_LOADER_MODE reserved for firmware upgrade only

5.18.4.6 SUSI_IMC_CONTROLPANEL_LoadDefaultValue

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_CONTROLPANEL_LoadDefaultValue** (void);
Android	short **SUSI_IMC_CONTROLPANEL_LoadDefaultValue**()

Description:

Load the default value on the TREK-303H/L firmware.

Parameters:

None

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

Notify: Call this function will reset all value which related to brightness and hoykey LED setting

5.18.4.7 SUSI_IMC_CONTROLPANEL_TEMPERATURE_GetValue

Syntax:

Linux / Windows	USHORT **SUSI_IMC_CONTROLPANEL_TEMPERATURE_GetValue** (OUT float *val);
Android	short **SUSI_IMC_CONTROLPANEL_TEMPERATURE_GetValue**(float[] val)

Description:

Return degree Celsius.

Parameters:

val [out]

Degree Celsius

Linux / Windows

Pointer of variable used to store returned value.

Android:

An allocated float array of size 1 for storing returned value.

The returned value will be stored at index 0.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-306D Linux / Windows

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

Notify: Only Support TREK-306D

5.19 LIGHT SENSOR API FUNCTIONS

5.19.1 Usage Overview

The CONTROL PANEL, LIGHT SENSOR, BRIGHTNESS, HOTKEY API modules are used to control the external smart display device (ex: TREK-306, TREK-303). The LIGHT SENSOR module provides API to get light sensor value.

The following section describes the usage of APIs.

5.19.1.1 Basic Usage

This section describes the basic usage of Light Sensor API module.

Windows/Linux

The CONTROL PANEL should be initialized first before you calling any APIs of the LIGHT SENSOR, BRIGHTNESS, HOTKEY API modules.

Please refer to the CONTROLPANEL usage for how to initialize.

Android

The class for LIGHT SENSOR APIs for Android is **mrm.client.VCILServiceClient**.

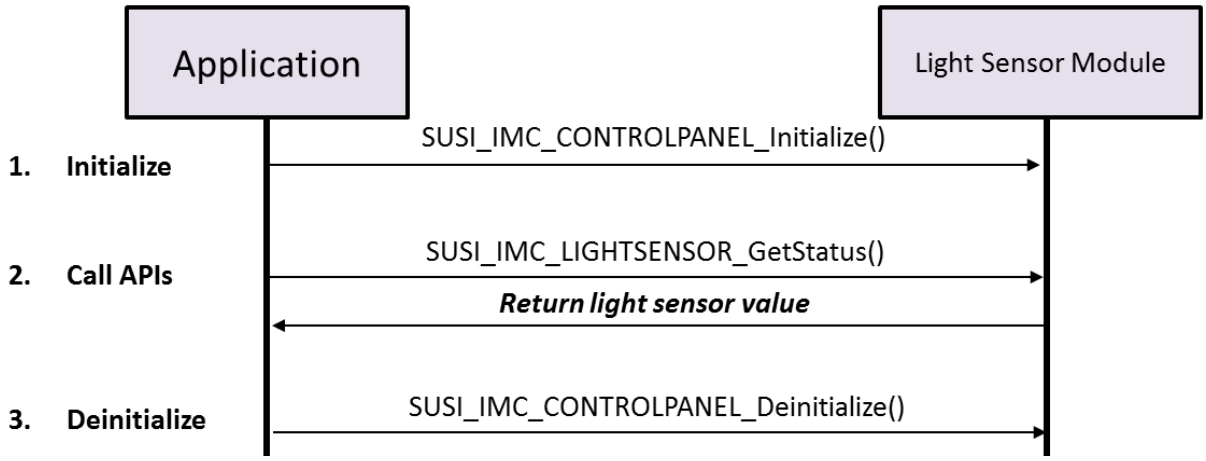
The APP should be connected to SDP service before you calling any APIs of the CONTROL PANEL, LIGHT SENSOR, BRIGHTNESS, HOTKEY API modules.

Please refer to the SDP SRV Usage Overview for basic usage.

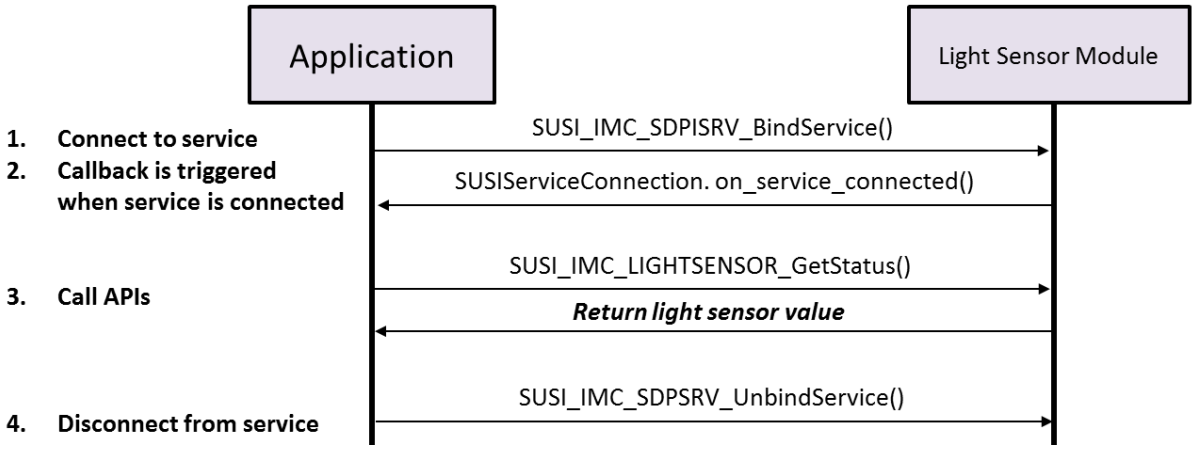
5.19.1.2 Light sensor status

You can use SUSI_IMC_LIGHTSENSOR_GetStatus() to check current light sensor value. The following shows the usage of API.

Windows/Linux



Android


Application
Light Sensor Module
1. Connect to service
2. Callback is triggered when service is connected
3. Call APIs
4. Disconnect from service
SUSI_IMC_SDPISRV_BindService()
SUSIServiceConnection. on_service_connected()
SUSI_IMC_LIGHTSENSOR_GetStatus()
Return light sensor value
SUSI_IMC_SDPSRV_UnbindService()

5.19.2 APIs

5.19.2.1 SUSI_IMC_LIGHTSENSOR_GetLibVersion

Syntax:

CE / Windows	USHORT **SUSI_IMC_LIGHTSENSOR_GetLibVersion**(OUT BYTE version[18]);

Description:

Get current version from the light sensor library.

Parameters:

version [out]

Point to read buffer that specifies the light sensor library's current version.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 CE

TREK-570 Windows

TREK-572 Windows

TREK-674 Windows

Remark:

The version buffer size must be greater than 18 bytes.

5.19.2.2 SUSI_IMC_LIGHTSENSOR_GetStatus

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_LIGHTSENSOR_GetStatus**(OUT PBYTE status);
Android	short **SUSI_IMC_LIGHTSENSOR_GetStatus**(int[] lightValue)

Description:

Get the normalized value of Light Sensor.

Parameters:

status [out]

Specifies the normalized value.

Linux / Windows

Pointer of variable used to store returned value.

Android:

An allocated array of size 1 for storing returned value.

The returned value will be stored at index 0.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.20 TEMPERATURE SENSOR API FUNCTIONS

5.20.1 Usage Overview

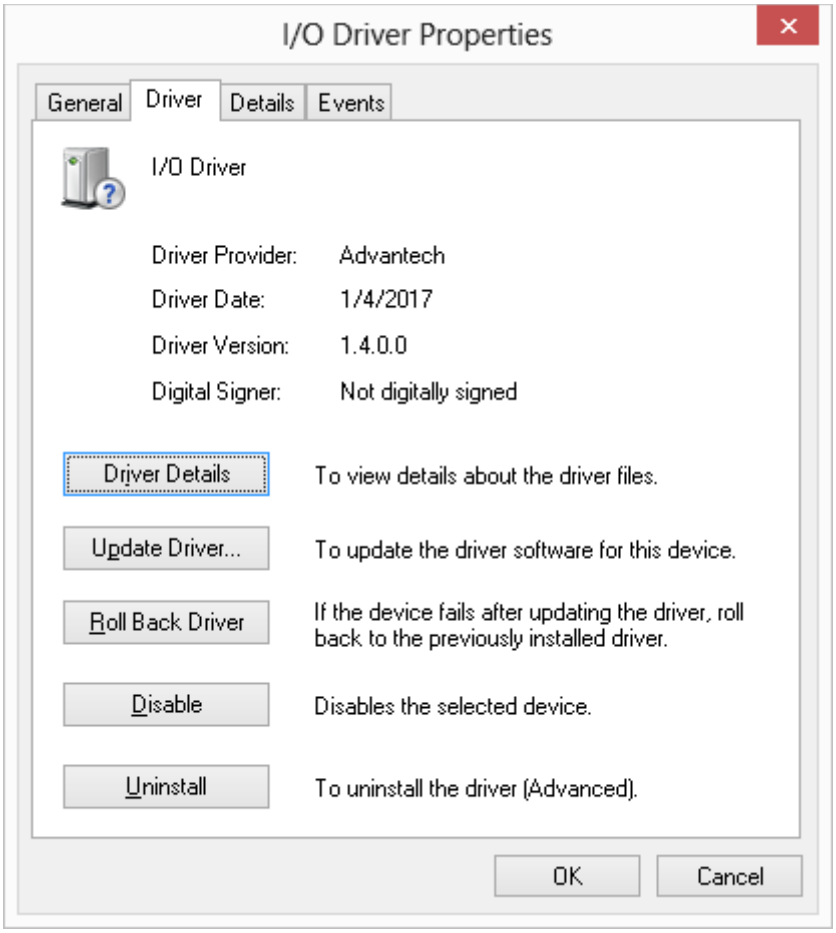
The Temperature Sensor provides API to get CPU and System temperature value.

The following section describes the usage of APIs.

5.20.1.1 Windows/Linux

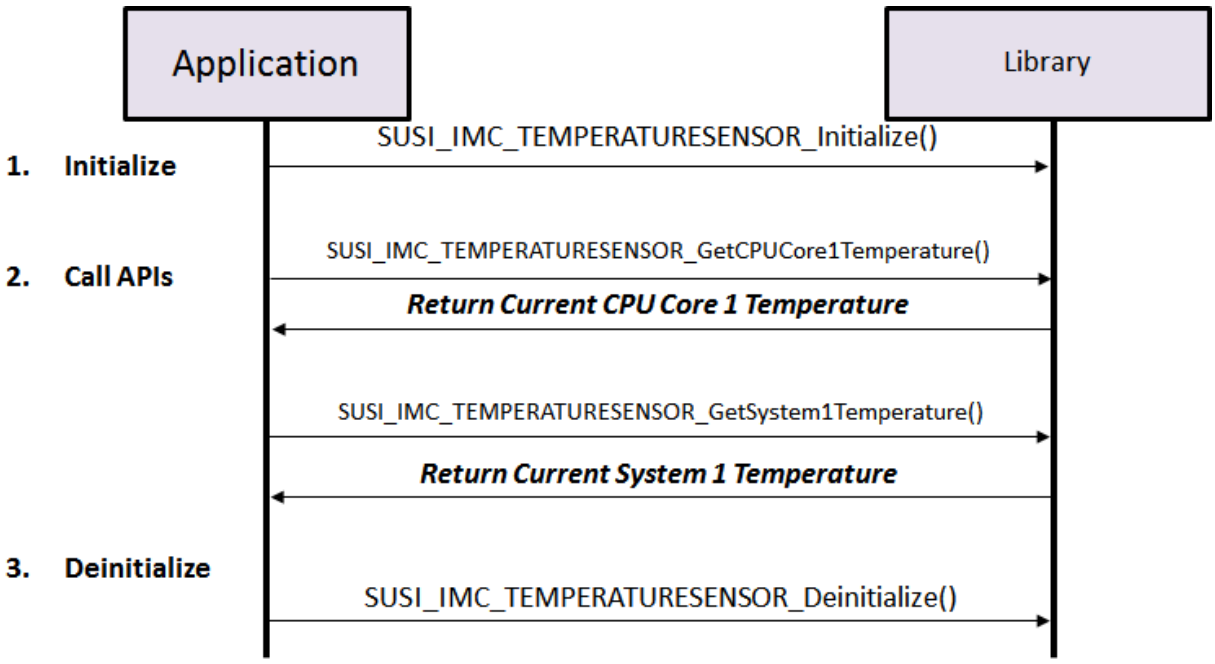
For using Temperature Sensor API, your I/O Driver version should be bigger than 1.4.0.0 Please check the driver version following below

Device Manager > External Devices > I/O Driver > (Right Click) Properties > Driver Page > Driver Version 1.4.0.0



If your I/O Driver not installed. SUSI_IMC_TEMPERATURESENSOR_Initialize() will return open fail. If your I/O Driver too old, the SUSI_IMC_TEMPERATURESENSOR_GetCPUCore1Temperature will return IMC_ERR_OPERATION_FAILED(0xC00A).

For all Temperature Sensor APIs, you should call SUSI_IMC_TEMPERATURESENSOR_Initialize() before using the other APIs. To stop the Temperature Sensor module, you must use SUSI_IMC_TEMPERATURESENSOR_Deinitialize() to deinitialize.



5.20.2 APIs

5.20.2.1 SUSI_IMC_TEMPERATURESENSOR_Initialize

Syntax:

Linux / Windows	USHORT **SUSI_IMC_TEMPERATURESENSOR_Initialize**();

Description:

Initialize the Temperature Sensor library.

Parameters:

None

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows

TREK-572 Linux / Windows

TREK-674 Linux / Windows

5.20.2.2 SUSI_IMC_TEMPERATURESENSOR_Deinitialize

Syntax:

Linux / Windows	USHORT **SUSI_IMC_TEMPERATURESENSOR_Deinitialize**(void);

Description:

Deinitialize the Temperature Sensor library.

Parameters:

None

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows

TREK-572 Linux / Windows

TREK-674 Linux / Windows

5.20.2.3 SUSI_IMC_TEMPERATURESENSOR_GetLibVersion

Syntax:

Windows	USHORT **SUSI_IMC_TEMPERATURESENSOR_GetLibVersion**(OUT char *version);

Description:

Get current library version from the temperature sensor library.

Parameters:

version [out]

Point to read buffer that specifies the temperature sensor library's current version.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Windows

TREK-572 Windows

TREK-674 Windows

Remark:

The version buffer size must be greater than 18 bytes.

5.20.2.4 SUSI_IMC_TEMPERATURESENSOR_GetCPUCore1Temperature

Syntax:

Linux / Windows	USHORT **SUSI_IMC_TEMPERATURESENSOR_GetCPUCore1Temperature**(OUT PBYTE celsius);

Description:

Get the CPU Core 1 temperature. The data obtained by ModelSpecific Register (MSR).

Parameters:

celsius [out]

Pointer of variable used to store returned value. The value represents Celsius

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows

TREK-572 Linux / Windows

TREK-674 Linux / Windows

5.20.2.5 SUSI_IMC_TEMPERATURESENSOR_GetCPUCore2Temperature

Syntax:

Linux / Windows	USHORT **SUSI_IMC_TEMPERATURESENSOR_GetCPUCore2Temperature**(OUT PBYTE celsius);

Description:

Get the CPU Core 2 temperature. The data obtained by ModelSpecific Register (MSR).

Parameters:

celsius [out]

Pointer of variable used to store returned value. The value represents Celsius

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows

TREK-572 Linux / Windows

TREK-674 Linux / Windows

5.20.2.6 SUSI_IMC_TEMPERATURESENSOR_GetSystem1Temperature

Syntax:

Linux / Windows	USHORT **SUSI_IMC_TEMPERATURESENSOR_GetSystem1Temperature**(OUT PBYTE celsius);

Description:

Get System 1 temperature. The data obtained by SuperIO Chip external Hardware Monitor sensor.

Parameters:

celsius [out]

Pointer of variable used to store returned value. The value represents Celsius

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows

TREK-572 Linux / Windows

TREK-674 Linux / Windows

5.21 BRIGHTNESS API FUNCTIONS

5.21.1 Usage Overview

The CONTROL PANEL, LIGHT SENSOR, BRIGHTNESS, HOTKEY API modules are used to control the external smart display device (ex: TREK-306, TREK-303). The BRIGHTNESS module provides APIs to/set get the brightness level and get/set duty cycle of each level.

The following section describes the usage of APIs.

5.21.1.1 Basic Usage

This section describes the basic usage of BRIGHTNESS API module.

Windows/Linux

The CONTROL PANEL should be initialized first before you calling any APIs of the LIGHT SENSOR, BRIGHTNESS, HOTKEY API modules.

Please refer to the CONTROLPANEL usage for how to initialize.

Android

The class for BRIGHTNESS APIs for Android is **mrm.client.VCILServiceClient**.

The APP should be connected to SDP service before you calling any APIs of the CONTROL PANEL, LIGHT SENSOR, BRIGHTNESS, HOTKEY API modules.

Please refer to the SDP SRV Usage Overview for basic usage.

5.21.1.2 Brightness Setting

The brightness settings are divided into two parts - brightness level and duty cycle:

- **Brightness Level Settings**

You can set the max, min and current brightness level of the smart display device by using SUSI_IMC_BRIGHTNESS_SetLevel(). The "current" level is used to assign the brightness level that the device currently displayed. The "max" and "min" level are used to set the upper and lower bounds of brightness level that can be adjusted from hard keys of device. You can also get the current settings by using SUSI_IMC_BRIGHTNESS_GetLevel().

- **Duty Cycle Settings**

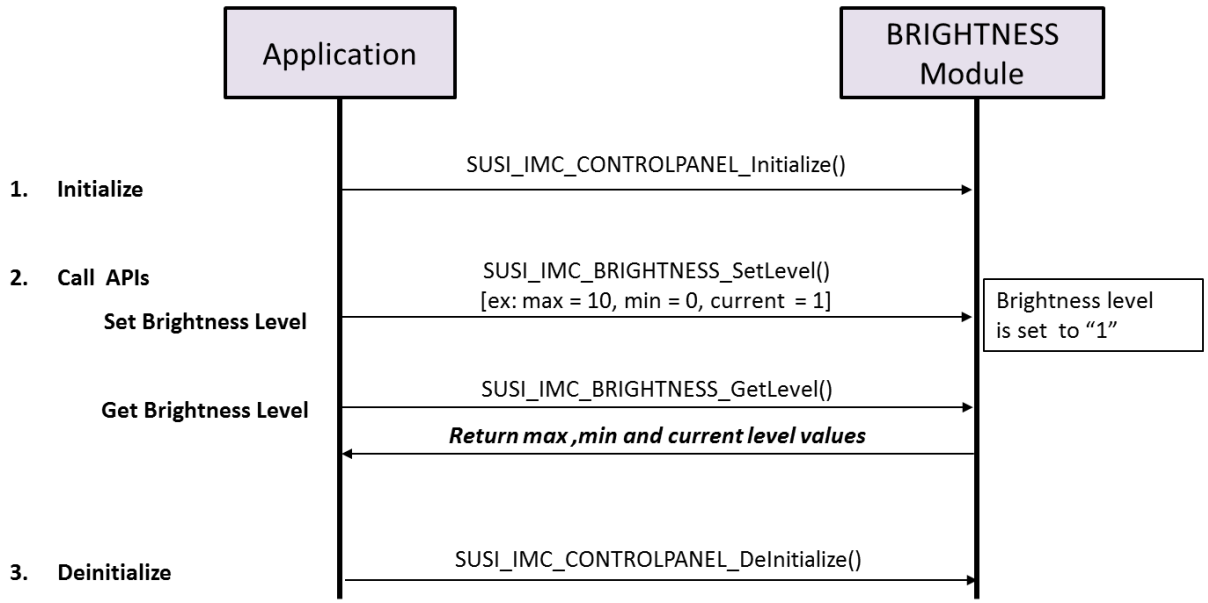
Each brightness level is set with a duty cycle value of back light. The duty cycle value is the represented in percentage of back light brightness. The default duty cycle of level 1 is 10% and the default duty cycle of level 10 is 100%.
You can customize the duty cycle value of each brightness level by using SUSI_IMC_BRIGHTNESS_SetDutyCycle() and get the settings by using SUSI_IMC_BRIGHTNESS_GetDutyCycle() .

Remark:

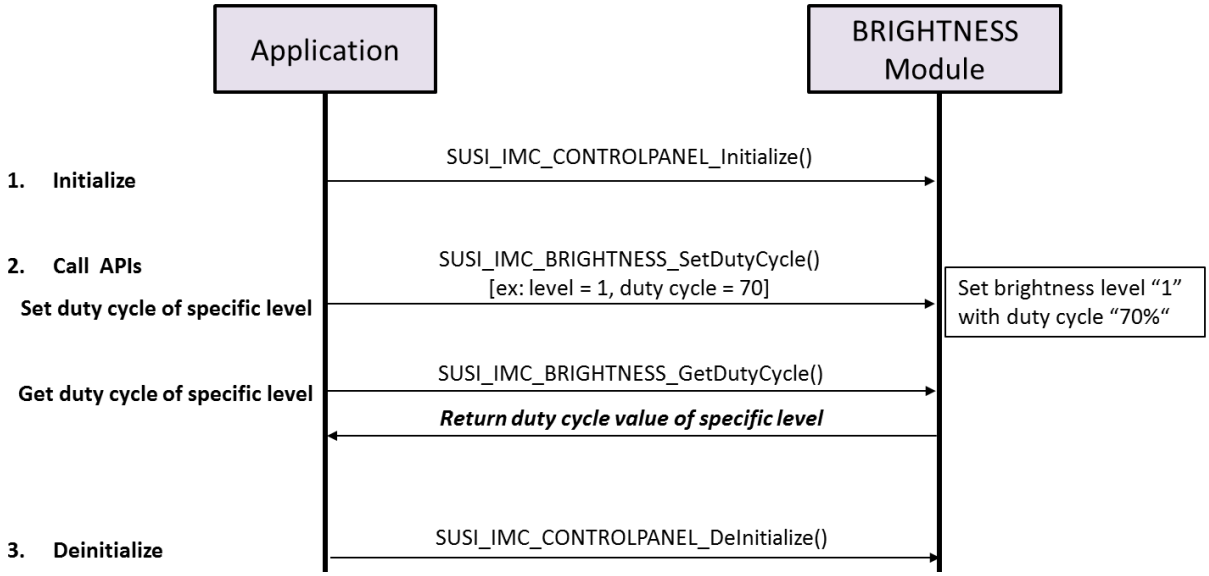
- The level 0 always equal to 0%. You can't change the brightness of level 0.
- Level Range is limited to 0-30. The current level can not out of the range.
- Duty Cycle Range is limited to 1-100. 100 represent the brightest.

Windows/Linux

- **Brightness Settings**

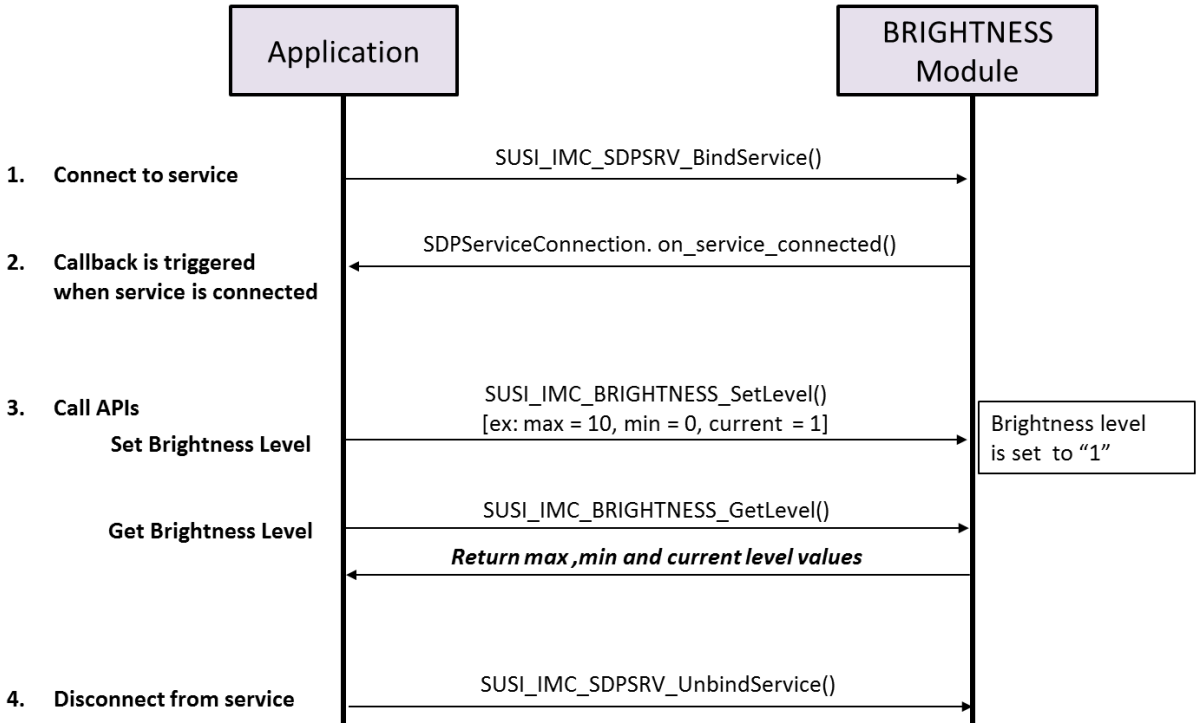


- **Duty Cycle Settings**

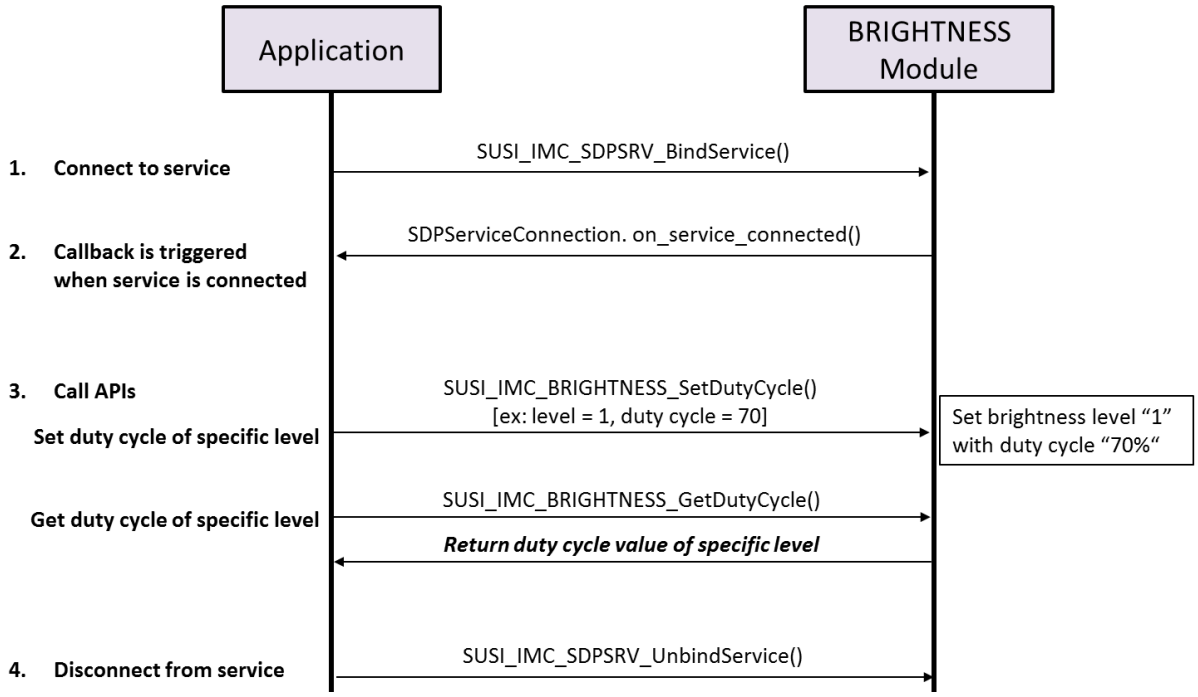


Android

- **Brightness Settings**



- **Duty Cycle Settings**



5.21.2 Structure/Classes

5.21.2.1 Windows/Linux

IMC_BRIGHTNESS_LEVEL Structure

Syntax:

```
typedef struct
{
        BYTE maximum;
        BYTE minimum;
        BYTE current;
}IMC_BRIGHTNESS_LEVEL,*PIMC_BRIGHTNESS_LEVEL;
```

Description:

This structure defines the brightness level.

Members:

maximum

Specifies the Maxima level is from 0 to 30.

minimum

Specifies the Minima level is from 0 to 30.

current

Specifies the Current level is from 0 to 30.

Remark:

The No-Change Level is 31.

5.21.2.2 Android

IMC_BRIGHTNESS_LEVEL

class:

mrm.define.brightness.**IMC_BRIGHTNESS_LEVEL**

Fields:

Field Name	Type	Input/Output	Comment
maximum	byte	in / out	The maximum brightness level. 0 to 30.
minimum	byte	in / out	The minimum brightness level. 0 to 30.
current	byte	in / out	The current brightness level. 0 to 30.

5.21.3 APIs

5.21.3.1 SUSI_IMC_BRIGHTNESS_SetDutyCycle

Syntax:

Linux / Windows	USHORT **SUSI_IMC_BRIGHTNESS_SetDutyCycle** (IN BYTE level, IN BYTE duty_cycle);
Android	short **SUSI_IMC_BRIGHTNESS_SetDutyCycle**(byte level, byte dutyCycle)

Description:

Set the duty cycle of various levels with Brightness.

Parameters:

level [in]

The level ID, from 1 to 30.

duty_cycle [in]

Specifies the duty cycle of Brightness, from 1 to 100.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

All levels can be modified, except level 0.

5.21.3.2 SUSI_IMC_BRIGHTNESS_GetDutyCycle

Syntax:

Linux / Windows	USHORT **SUSI_IMC_BRIGHTNESS_GetDutyCycle** (IN BYTE level, OUT PBYTE duty_cycle);
Android	short **SUSI_IMC_BRIGHTNESS_GetDutyCycle**(byte level, byte[] dutyCycle)

Description:

Get the duty cycle of various levels with Brightness.

Parameters:

level [in]

The level ID, from 1 to 30.

duty_cycle [out]

Duty cycle of Brightness.

Linux / Windows

Pointer of variable used to store returned value.

Android:

An allocated array of size 1 for storing returned value.

The returned value will be stored at index 0.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.21.3.3 SUSI_IMC_BRIGHTNESS_SetLevel

Syntax:

Linux / Windows	USHORT **SUSI_IMC_BRIGHTNESS_SetLevel** (IN PIMC_BRIGHTNESS_LEVEL level);
Android	short **SUSI_IMC_BRIGHTNESS_SetLevel**(IMC_BRIGHTNESS_LEVEL level)

Description:

Set the variety level of Brightness.

Parameters:

level [in]

Points to a variable that specifies the Brightness level.

Linux / Windows

A IMC_BRIGHTNESS_LEVEL structure storing the values to be set.

Android:

An instance of IMC_BRIGHTNESS_LEVEL storing the values to be set.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

- Set the SUSI_BRIGHTNESS_NON_CHANGE(31) if this member not changes.
  For example, if you want change the current level only. you need set the IMC_BRIGHTNESS_LEVEL.maximum and IMC_BRIGHTNESS_LEVEL.minimum SUSI_BRIGHTNESS_NON_CHANGE(31).

- You should not set minimum, maximum and current level at int the same call or the API may returns error.
  You should set minimum and maximum first, and then set the current level within the range.
  ex:
  The original level range is 1~10.
  If You want to set range to 2~20 and set current range to 5, you should make the following calls sequentially.
  - SUSI_IMC_BRIGHTNESS_SetLevel (level); //Where level.minimum = 2, level.minimum = 20,

level.current = 31

- SUSI_IMC_BRIGHTNESS_SetLevel (level); //Where level.minimum = 2, level.minimum = 20, level.current = 5

5.21.3.4 SUSI_IMC_BRIGHTNESS_GetLevel

Syntax:

Linux / Windows	USHORT **SUSI_IMC_BRIGHTNESS_GetLevel** (OUT PIMC_BRIGHTNESS_LEVEL level);
Android	short **SUSI_IMC_BRIGHTNESS_GetLevel**(IMC_BRIGHTNESS_LEVEL level)

Description:

Get the variety level of Brightness.

Parameters:

level [out]

Points to a variable that specifies the Brightness level.

Windows:

A pointer to IMC_BRIGHTNESS_LEVEL structure for storing returned values

Android:

Instance of IMC_BRIGHTNESS_LEVEL for storing returned value;

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.22 HOTKEY API FUNCTIONS

5.22.1 Usage Overview

The CONTROL PANEL, LIGHT SENSOR, BRIGHTNESS, HOTKEY API modules are used to control the external smart display device (ex: TREK-306, TREK-303). The HOTKEY module provides API to get hotkey press status and to set/get duty cycle of hotkey LED.

The following section describes the usage of APIs.

5.22.1.1 Basic Usage

This section describes the basic usage of HOTKEY API module.

Windows/Linux

The CONTROL PANEL should be initialized first before you calling any APIs of the LIGHT SENSOR, BRIGHTNESS, HOTKEY API modules.

Please refer to the CONTROLPANEL usage for how to initialize.

Android

The class for HOTKEY APIs for Android is **mrm.client.VCILServiceClient**.

The APP should be connected to SDP service before you calling any APIs of the CONTROL PANEL, LIGHT SENSOR, BRIGHTNESS, HOTKEY API modules.

Please refer to the SDP SRV Usage Overview for basic usage.

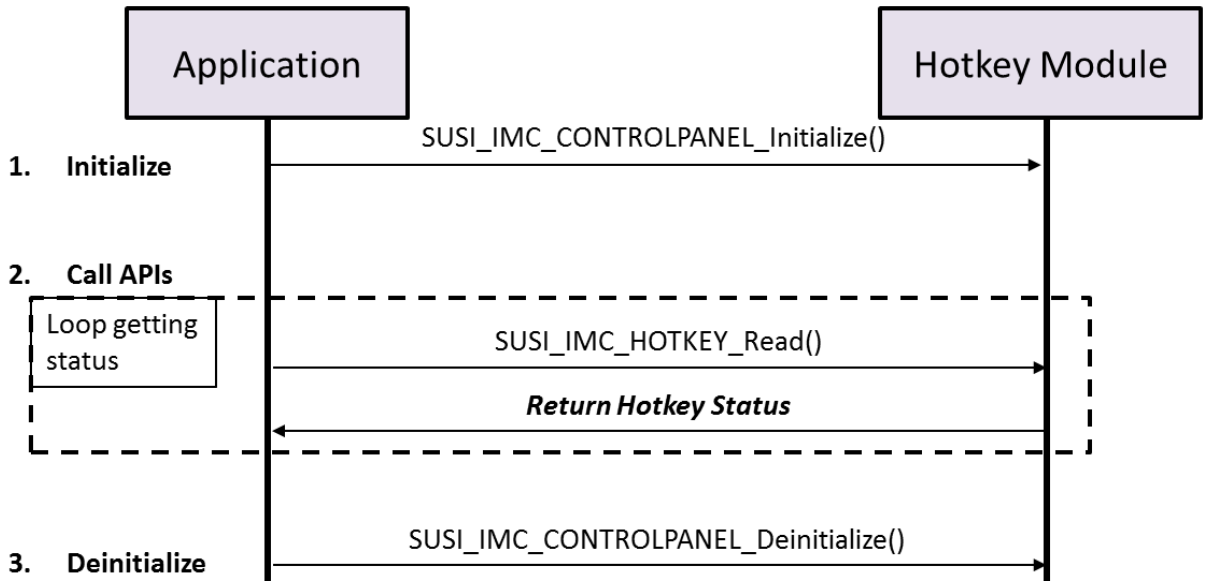
5.22.1.2 Hotkey Status Getting

This section describes how to get hotkey press status.

Windows/Linux

You implement your hotkey status reading application in either **Polling** or **Event Handling** style.

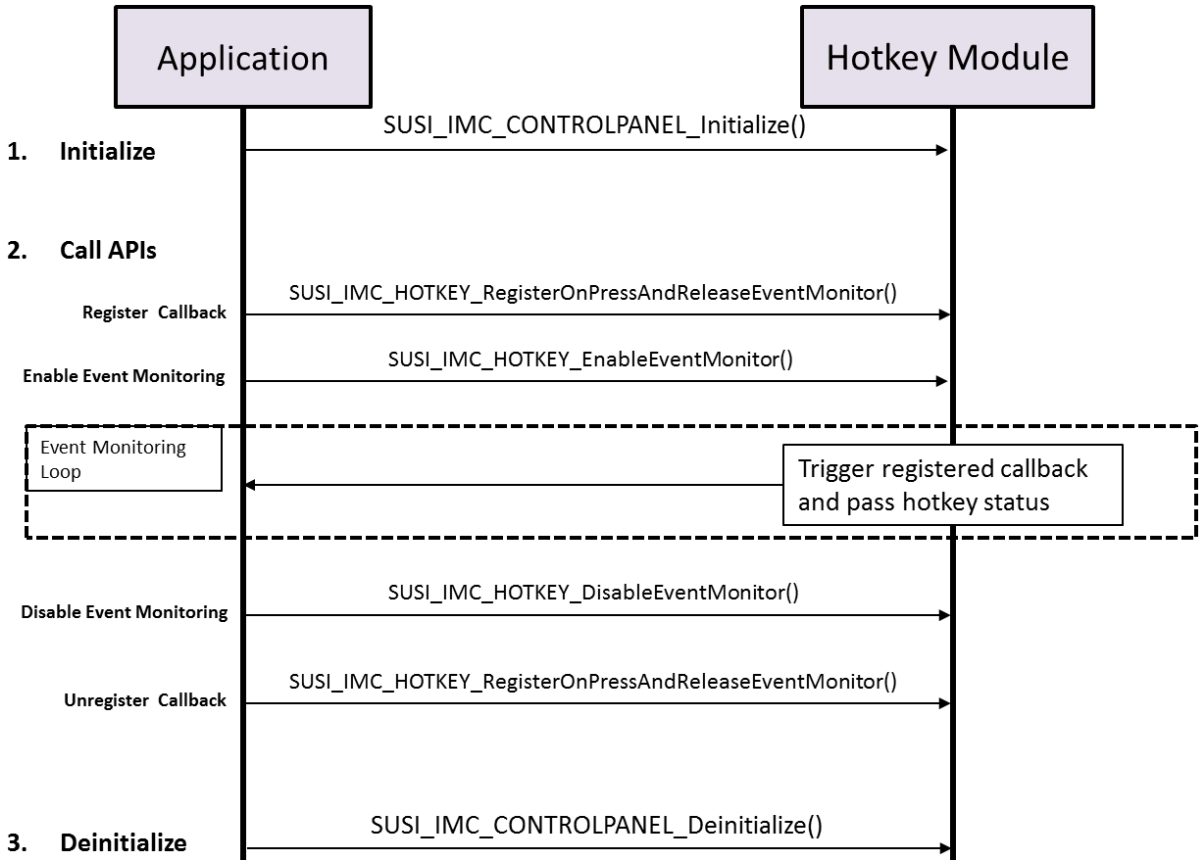
- **Polling:**



The most simple way to get hotkey status is to hold a loop in your APP and keep calling SUSI_IMC_HOTKEY_Read() .

This methods is relative simple but may result in unnecessary power consuming.

- **Event Handling:**



To get hotkey status in event handling style, you need to call SUSI_IMC_HOTKEY_RegisterOnPressAndReleaseEventMonitor() first to register a callback to HOYKEY API module and call SUSI_IMC_HOTKEY_EnableEventMonitor() to enable event monitoring in HOYKEY API module.

When hotkey status changes(i.e. pressed or released), HOYKEY API module triggers the registered callback and pass the hotkey status to client APP.

If you do not want to monitor hotkey status event any more, you should call SUSI_IMC_HOTKEY_DisableEventMonitor() to disable event monitoring in HOYKEY API module, then call SUSI_IMC_HOTKEY_ClearOnPressEventMonitor() to unregister callback.

The advantage of getting hotkey status in event handling style is that APP's process only works when hotkey status changed, which relatively costs less system resources and power.
The disadvantage is that there would be some overhead of event handling.

Please refer to the sample code for implementation details.

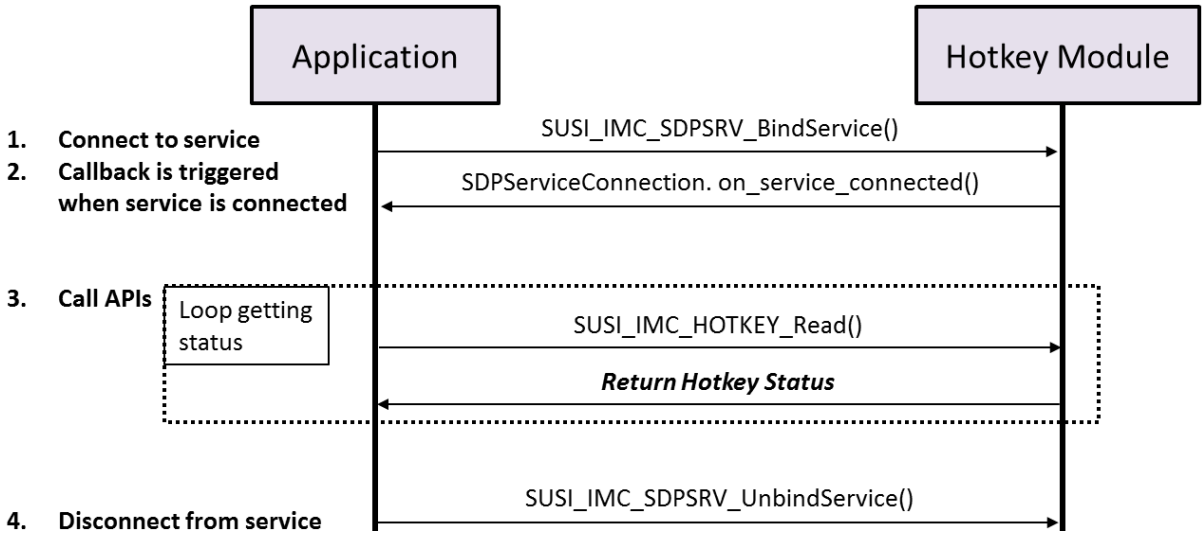
Remark:

SUSI_IMC_HOTKEY_DisableEventMonitor() **MUST** be called before SUSI_IMC_HOTKEY_ClearOnPressEventMonitor() otherwise it will result in unexpected erroneous behavior.

Android

You implement your hotkey status reading application in either **Polling** or **Event Handling** style.

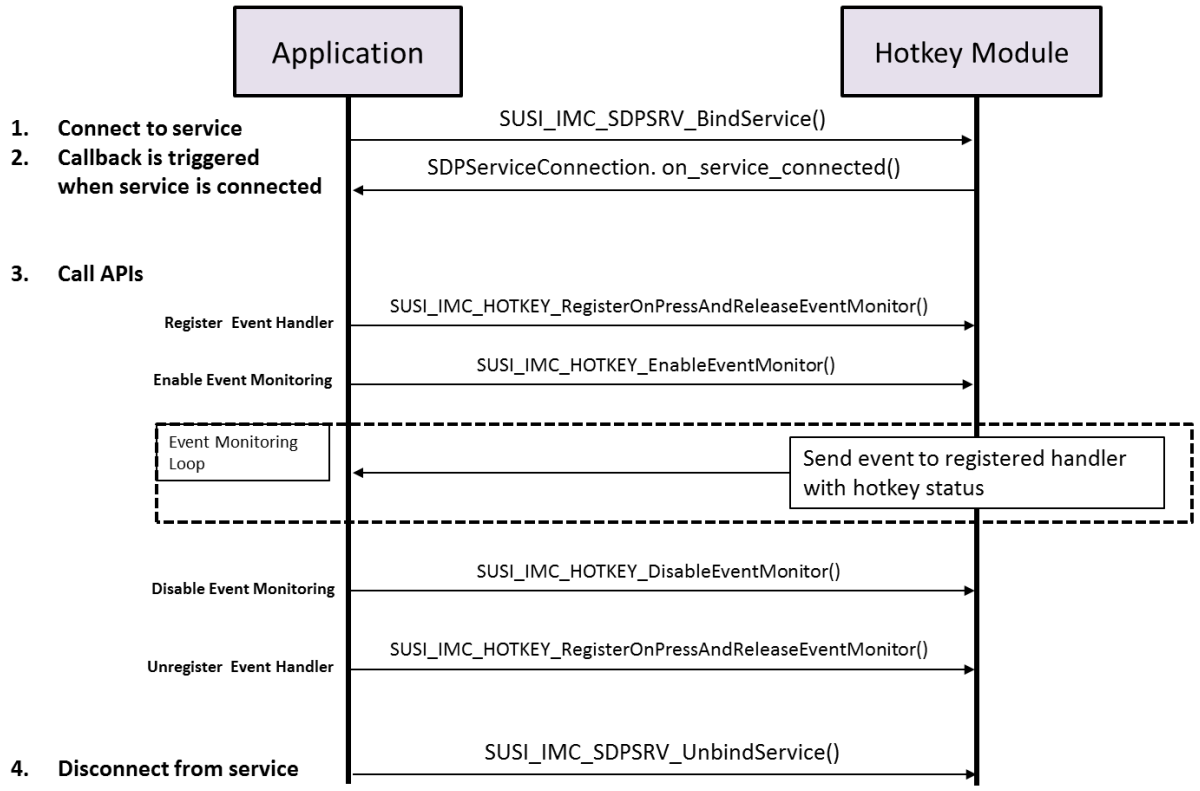
- **Polling:**



The most simple way to get hotkey status is to hold a loop in your APP and keep calling SUSI_IMC_HOTKEY_Read() .

This methods is relative simple but may result in unnecessary power consuming.

- **Event Handling:**



TREK SDK leverage the Android Handler mechanism to inform hotkey press event.

To get hotkey status in event handling style, you need to create an instance of Handler and call SUSI_IMC_HOTKEY_RegisterOnPressAndReleaseEventMonitor() first to register the Handler instance to HOTKEY API module and call SUSI_IMC_HOTKEY_EnableEventMonitor() to enable event monitoring in HOYKEY API module.

When hotkey status changes(i.e. pressed or released), SDP service trigger event the and send event with the hotkey status to the registered handler instance.

If you do not want to monitor hotkey status event any more, you should call SUSI_IMC_HOTKEY_DisableEventMonitor() to disable event monitoring in HOYKEY API module, then call SUSI_IMC_HOTKEY_ClearOnPressEventMonitor() to unregister handler.

The advantage of getting hotkey status in event handling style is that APP's process only works when hotkey status changed, which relatively costs less system resources and power.
The disadvantage is that there would be some overhead of event handling.

Please refer to the sample code for implementation details.

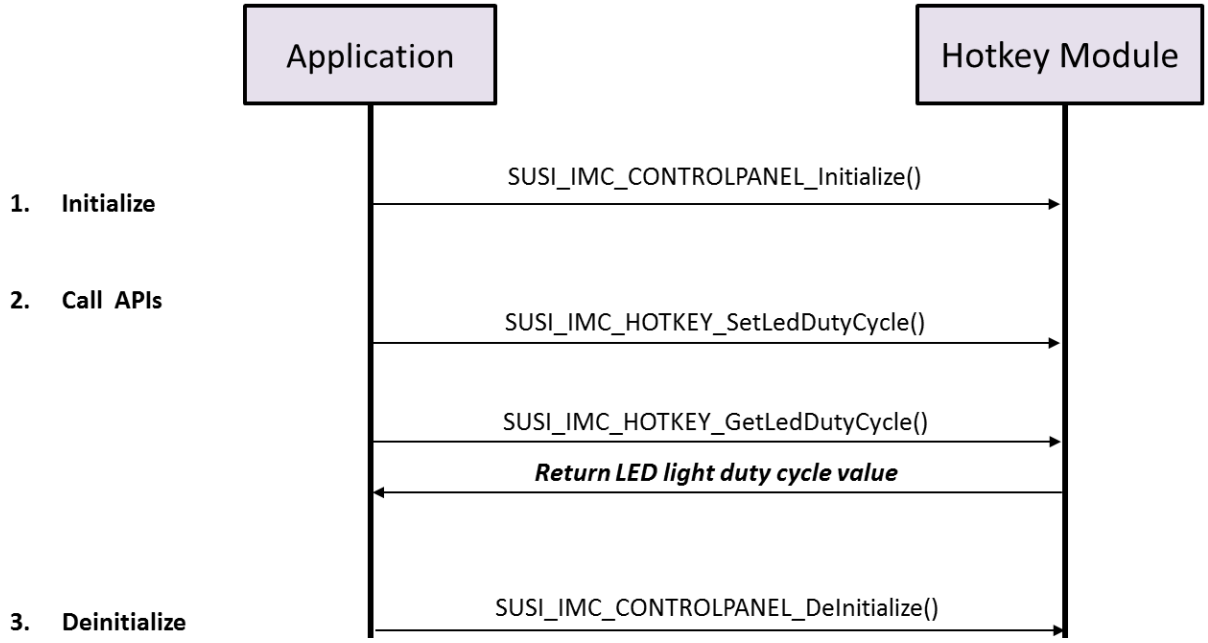
Remark:
SUSI_IMC_HOTKEY_DisableEventMonitor() **MUST** be called before SUSI_IMC_HOTKEY_ClearOnPressEventMonitor() otherwise it will result in unexpected erroneous

behavior.

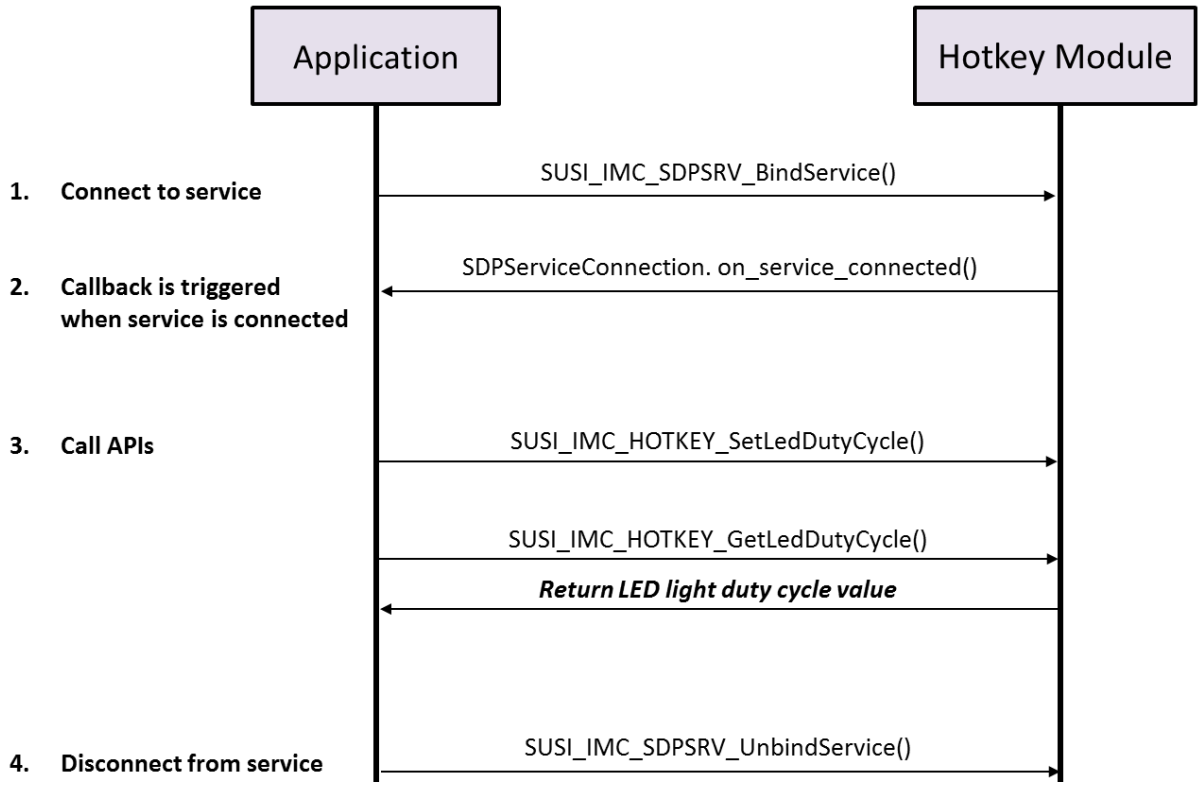
5.22.1.3 Hotkey LED Duty Cycle Setting

TREK SDK provides APIs to get/set the LED duty cycle of the hotkeys. You can use SUSI_IMC_HOTKEY_SetLedDutyCycle() to set and SUSI_IMC_HOTKEY_GetLedDutyCycle() to get the duty cycle.

Windows/Linux



Android


Application
Hotkey Module
1. Connect to service
SUSI_IMC_SDPSRV_BindService()
2. Callback is triggered when service is connected
SDPServiceConnection. on_service_connected()
3. Call APIs
SUSI_IMC_HOTKEY_SetLedDutyCycle()
SUSI_IMC_HOTKEY_GetLedDutyCycle()
Return LED light duty cycle value
4. Disconnect from service
SUSI_IMC_SDPSRV_UnbindService()

5.22.2 Structure/Classes

5.22.2.1 Windows/Linux

IMC_HOTKEY_MSG_OBJECT Structure

Syntax:

```
typedef struct
{
        BYTE buf[SUSI_HOTKEY_BUFFER_SIZE];
} IMC_HOTKEY_MSG_OBJECT, *PIMC_HOTKEY_MSG_OBJECT
```

Description:

This structure defines the hotkey message.

Members:

buf

data buffer of hotkey

5.22.2.2 Android

IMC_HOTKEY_MSG_OBJECT

class:

mrm.define.hotkey.**IMC_HOTKEY_MSG_OBJECT**

Fields:

Field Name	Type	Input/Output	Comment
buf	byte[]	out	The array which store status of all hotkeys. Value definition: 0: Button released. 1: Buttons pressed.

5.22.3 APIs

5.22.3.1 SUSI_IMC_HOTKEY_Read

Syntax:

Linux / Windows	USHORT **SUSI_IMC_HOTKEY_Read** (OUT PIMC_HOTKEY_MSG_OBJECT object);
Android	short **SUSI_IMC_HOTKEY_Read**(IMC_HOTKEY_MSG_OBJECT object)

Description:

Reads the status of all hotkeys

Parameters:

object [out]

Linux / Windows

A pointer to IMC_HOTKEY_MSG_OBJECT structure storing the values to be set.

Android:

An instance of IMC_HOTKEY_MSG_OBJECT storing the values to be set.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

The key value is equal to 1 when the key down.

5.22.3.2 SUSI_IMC_HOTKEY_RegisterOnPressAndReleaseEventMonitor (Windows/Linux)

Syntax:

Linux / Windows	USHORT **SUSI_IMC_HOTKEY_RegisterOnPressAndReleaseEventMonitor** (IN HOTKEY_OnPressAndRelease_Funptr fptr);

Description:

Register the hotkey routine of callback function to event monitor. The callback will be called when hotkey status changed.

Parameters:

fptr [in]

Linux / Windows

A pointer to callback functions .

Android:

An instance of Handler.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.22.3.3 SUSI_IMC_HOTKEY_ClearOnPressEventMonitor (Windows/Linux)

Syntax:

Linux / Windows	USHORT **SUSI_IMC_HOTKEY_ClearOnPressEventMonitor** (void);

Description:

This function clears the setting of event monitor.

Parameters:

None

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

Before clear the callback function you should calling **SUSI_IMC_HOTKEY_DisableEventMonitor()** to disable hotkey callback function otherwise the system may crash since program access the invalid memory.

5.22.3.4 SUSI_IMC_HOTKEY_RegisterOnPressAndReleaseEventMonitor (Android)

Syntax:

Android	short **SUSI_IMC_HOTKEY_RegisterOnPressAndReleaseEventMonitor**(Handler handler)

Description:

Register a instance of Handler to HOTKEY API module. When hotkey status changed, the Handler.handleMessage() will be called and receives a Message object with "what" field of SDP_EVENT_ID_HOTKEY_PRESS_STATUS_CHANGED.

Parameters:

handler [in]

Android:

An instance of Handler.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Android

Remark:

None

5.22.3.5 SUSI_IMC_HOTKEY_ClearOnPressEventMonitor (Android)

Syntax:

Linux / Windows	USHORT **SUSI_IMC_HOTKEY_ClearOnPressEventMonitor** (void);

Description:

Unregister event handler.

Parameters:

None

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Android

Remark:

SUSI_IMC_HOTKEY_DisableEventMonitor() **MUST** be called before **SUSI_IMC_HOTKEY_ClearOnPressEventMonitor**(), otherwise it will result in unexpected erroneous behavior.
Please refer to the usage.

5.22.3.6 SUSI_IMC_HOTKEY_EnableEventMonitor

Syntax:

Linux / Windows	USHORT **SUSI_IMC_HOTKEY_EnableEventMonitor** (void);
Android	short **SUSI_IMC_HOTKEY_EnableEventMonitor**()

Description:

Start hotkey event monitoring.

Parameters:

None

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.22.3.7 SUSI_IMC_HOTKEY_DisableEventMonitor

Syntax:

Linux / Windows	USHORT **SUSI_IMC_HOTKEY_DisableEventMonitor** (void);
Android	short **SUSI_IMC_HOTKEY_DisableEventMonitor**()

Description:

Stop hotkey event monitoring.

Parameters:

None

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.22.3.8 SUSI_IMC_HOTKEY_SetLedDutyCycle

Syntax:

Linux / CE / Windows	USHORT **SUSI_IMC_HOTKEY_SetLedDutyCycle** (IN BYTE duty_cycle);
Android	short **SUSI_IMC_HOTKEY_SetLedDutyCycle**(byte dutyCycle)

Description:

Set the Hotkey led light duty cycle.

Parameters:

duty_cycle [in]

From 0 to 100.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

None

5.22.3.9 SUSI_IMC_HOTKEY_GetLedDutyCycle

Syntax:

Linux / Windows	USHORT **SUSI_IMC_HOTKEY_GetLedDutyCycle** (OUT PBYTE duty_cycle);
Android	short **SUSI_IMC_HOTKEY_GetLedDutyCycle**(byte[] dutyCycle)

Description:

Get the hotkey led light duty cycle.

Parameters:

duty_cycle [out]

From 0 to 100.

Linux / Windows

Pointer of variable used to store returned value.

Android:

An allocated array of size 1 for storing returned value.

The returned value will be stored at index 0.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-570 Linux / Windows / Android

TREK-572 Linux / Windows

TREK-674 Linux / Windows

Remark:

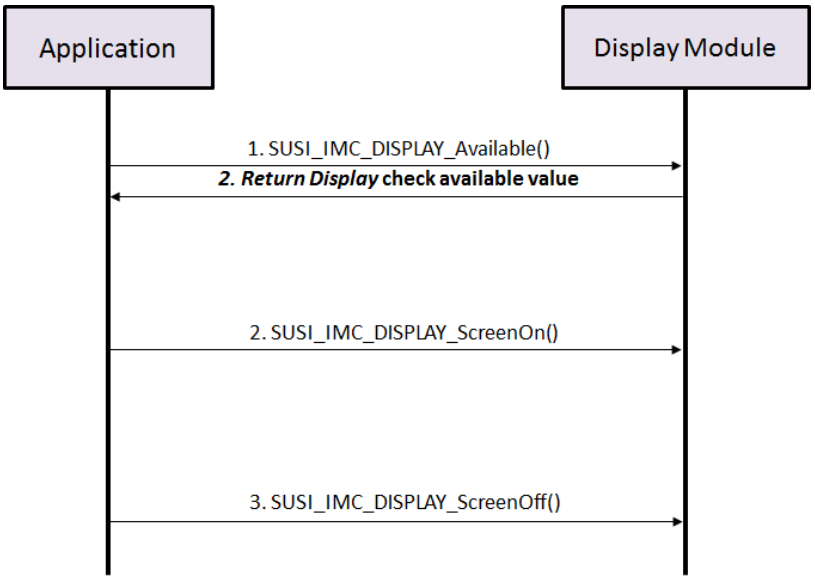
None

5.23 DISPLAY

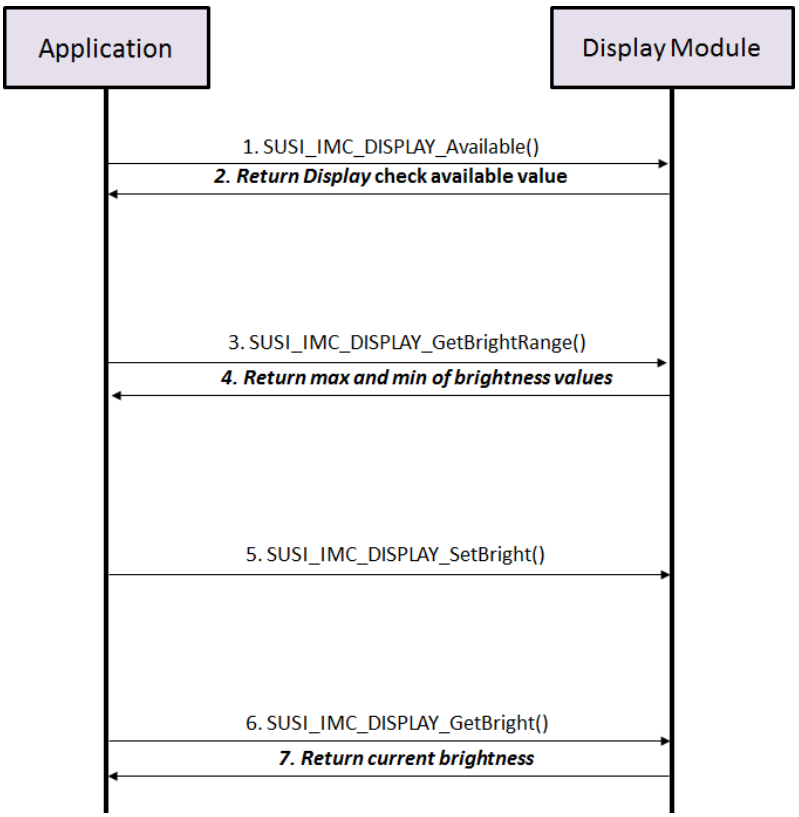
5.23.1 Usage Overview

5.23.1.1 Windows/Linux

Display module provides a way to control display screen and brightness. Use **SUSI_IMC_DISPLAY_Available()** to get display driver the feature support information. If display driver support all feature, then you can use **SUSI_IMC_DISPLAY_ScreenOn()** / **SUSI_IMC_DISPLAY_ScreenOff()** to turn on/off the display screen.



You also can use **SUSI_IMC_DISPLAY_GetBrightRange()** to check the level of brightness. Then, call **SUSI_IMC_DISPLAY_SetBright()** to set the brightness. Call **SUSI_IMC_DISPLAY_GetBright()** to get current brightness.


Application
Display Module
1. SUSI_IMC_DISPLAY_Available()
2. Return Display check available value
3. SUSI_IMC_DISPLAY_GetBrightRange()
4. Return max and min of brightness values
5. SUSI_IMC_DISPLAY_SetBright()
6. SUSI_IMC_DISPLAY_GetBright()
7. Return current brightness

5.23.2 Structure/Classes

5.23.2.1 Windows/Linux

IMC_DISPLAY_GET_BRIGHT_RANGE Structure

Syntax:

```
typedef struct
{
        BYTE *minimum;
            BYTE *maximum;
            BYTE *stepping;
}IMC_DISPLAY_GET_BRIGHT_RANGE,*PIMC_DISPLAY_GET_BRIGHT_RANGE;
```

Description:

This structure defines the bright range .

Members:

minimum [out]

Pointer to a variable to get the minimum brightness value.

maximum [out]

Pointer to a variable to get the maximum brightness value.

stepping [out]

Pointer to a variable to get the step of brightness up and down.

5.23.3 APIs

5.23.3.1 SUSI_IMC_DISPLAY_GetLibVersion

Syntax:

CE	USHORT **SUSI_IMC_DISPLAY_GetLibVersion**(OUT BYTE version[18]);

Description:

Get current version from the display library.

Parameters:

version [out]

Point to read buffer that specifies the display library's current version.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 CE

Remark:

The version buffer size must be greater than 18 bytes.

5.23.3.2 SUSI_IMC_DISPLAY_Available

Syntax:

Linux / CE	USHORT **SUSI_IMC_DISPLAY_Available**(void);

Description:

Check if display driver is available and also get the feature support information.

Parameters:

None

Returns:

- Value range
  IMC_ERR_DISPLAY_FUNCTION_OK_PLATFORM_NOT_SUPPORT_ALL - The function succeeds; the platform does not support any APIs.
  IMC_ERR_DISPLAY_FUNCTION_OK_ONLY_BRIGHTNESS - The function succeeds; the platform supports only brightness APIs.
  IMC_ERR_DISPLAY_FUNCTION_OK_ONLY_SCREEN_CONTROL - The function succeeds; the platform supports only screen on/off APIs.
  IMC_ERR_DISPLAY_FUNCTION_OK_SUPPORT - The function succeeds; the platform supports all APIs.
- Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

Remark:

None

5.23.3.3 SUSI_IMC_DISPLAY_ScreenOn

Syntax:

Linux / CE	USHORT **SUSI_IMC_DISPLAY_ScreenOn**(void);

Description:

Turn on display signal.

Parameters:

None

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

Remark:

None

5.23.3.4 SUSI_IMC_DISPLAY_ScreenOff

Syntax:

Linux / CE	USHORT **SUSI_IMC_DISPLAY_ScreenOff(**void);

Description:

Turn off display signal.

Parameters:

None

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

Remark:

None

5.23.3.5 SUSI_IMC_DISPLAY_GetBrightRange

Syntax:

Linux / CE	USHORT **SUSI_IMC_DISPLAY_GetBrightRange**(PIMC_DISPLAY_GET_BRIGHT_RANGE parm);

Description:

Get the step, minimum and maximum values in brightness adjustment.

Parameters:

parm [in/out]

Point to read buffer that specifies the information of display.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

Remark:

None

5.23.3.6 SUSI_IMC_DISPLAY_GetBright

Syntax:

Linux / CE	USHORT **SUSI_IMC_DISPLAY_GetBright**(BYTE *brightness);

Description:

Get the current panel brightness.

Parameters:

brightness [out]

Pointer to a variable in which this function returns the brightness.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

Remark:

None

5.23.3.7 SUSI_IMC_DISPLAY_SetBright

Syntax:

Linux / CE	USHORT **SUSI_IMC_DISPLAY_SetBright**(BYTE brightness);

Description:

Set current panel brightness.

Parameters:

brightness [in]

Specifies the brightness value to be set.

Returns:

IMC_ERR_NO_ERROR - On success.

Otherwise see the error code list.

Supported Platforms:

TREK-520 Linux / CE

Remark:

None

6 RETURN CODE LIST

The following topics provide lists of TREK SDK error codes.
These values are defined in the SUSI_IMC_Def.h and the SUSI_IMC_Error.h header files.

6.1 GENERAL ERROR CODES

Code	Value	Description
IMC_ERR_NO_ERROR	0xC000	The Operation was successful.
IMC_ERR_INVALID_POINTER	0xC001	The specified pointer is not a valid pointer.
IMC_ERR_INVALID_ARGUMENT	0xC002	One or more of the supplied parameters could not be properly interpreted.
IMC_ERR_UNSUPPORT	0xC003	The Operation not supported by the device.
IMC_ERR_OUT_OF_MEMORY	0xC004	No enough storage is available to complete this operation.
IMC_ERR_SDK_NOT_INIT	0xC005	Need to initialize instance first.
IMC_ERR_OBJ_INIT_FAILED	0xC006	Initialized instance failed.
IMC_ERR_INVALID_SIZE	0xC007	The input data size was out of range.
IMC_ERR_TIMEOUT	0xC008	The user-specified timeout value has expired.
IMC_ERR_FILE_OPEN	0xC009	Unable to open the file.
IMC_ERR_OPERATION_FAILED	0xC00A	The operation failed to complete.
IMC_ERR_DRIVER_OPEN	0xC00B	Failed to initialize the kernel driver.
IMC_ERR_UNKNOWN	0xC00B	Internal unknown error
IMC_ERR_SUSI_API	0xC00D	The SUSI driver internal error
IMC_ERR_FW_VERSION_TOO_OLD	0xC00E	The firmware is too old
IMC_ERR_INITIALIZED	0xC00F	The Library is initialized

IMC_ERR_ANDROID_JNI_EVENT_INIT_FAILED	0xC100	Failed to init event handle on service side(JNI).
IMC_ERR_ANDROID_JNI_EVENT_DEINIT_FAILED	0xC101	Failed to deinit event handle on service side(JNI).

IMC_ERR_ANDROID_JNI_EVENT_LISTENING_THREAD_CREATE_FAILED	0xC102	Service side error. JNI event listening thread create failed.
IMC_ERR_ANDROID_JNI_EVENT_LISTENING_THREAD_ALREADY_RUNNING	0xC103	Event listening thread in service is already running(JNI).
IMC_ERR_ANDROID_SERVICE_NULL_POINTER	0xC110	Null pointer error occurred on service side.
IMC_ERR_ANDROID_SERVICE_UNKNWON_EXCEPTION	0xC111	Unknown exception occurred on service side.
IMC_ERR_ANDROID_SERVICE_REMOTE_CALLBACK_LIST_NOT_FOUND	0xC112	Remote callback list not found on service side.
IMC_ERR_ANDROID_SERVICE_REMOTE_CALLBACK_REGISTER_FAILED	0xC113	Failed to register remote callback on service side.
IMC_ERR_ANDROID_SERVICE_REMOTE_CALLBACK_UNREGISTER_FAILED	0xC114	Failed to unregister remote callback on service side.
IMC_ERR_ANDROID_SERVICE_REMOTE_CALLBACK_UPDATE_FAILED	0xC115	Failed to update remote callback cookie on service side.
IMC_ERR_ANDROID_CLIENT_NULL_POINTER	0xC120	Null pointer error occurred on client side in Service Client API lib.
IMC_ERR_ANDROID_CLIENT_FAILED_TO_BIND_SERVICE	0xC121	Unable to connect client application context to the service.
IMC_ERR_ANDROID_CLIENT_SERVICE_ALREADY_CONNECTED	0xC122	Current client application context is already connected to the service.
IMC_ERR_ANDROID_CLIENT_SERVICE_DISCONNECTED	0xC123	Current client application context is not connected to the service.
IMC_ERR_ANDROID_CLIENT_REMOTE_EXCEPTION	0xC124	Remote exception received at client side. Failed to execute required task.
IMC_ERR_ANDROID_CLIENT_FAILED_TO_UNBIND_SERVICE	0xC125	Unable to disconnect client application context to the service.

6.2 CAN API ERROR CODE

Code	Value	Description
IMC_CAN_SET_BIT_TIMING_ERROR	0xC010	Failed to set the Bit-Timing of CAN.
IMC_CAN_RX_NOT_READY	0xC012	The RX buffer is empty.
IMC_CAN_TX_WRITE_ERR	0xC014	Failed to write a message to vehicle network.

6.3 VPM API ERROR CODE

Code	Value	Description
IMC_ERR_VPM_NOT_SET_ALARM	0xC040	The alarm wakeup not set.
IMC_ERR_VPM_COMPORT_BUSY	0xC041	COM port is currently busy, try later.
IMC_ERR_VPM_ERROR_WAKEUPSOURCE_FROM_VPM	0xC043	Error last wakeup source from VPM.
IMC_ERR_VPM_OPEN_VPM_COM_FAIL	0xC044	Open com of VPM fail.
IMC_ERR_VPM_UNKNOW_RETURN_VALUE_FROM_VPM	0xC045	Unknown return value form VPM. Please update your firmware
IMC_ERR_VPM_COMPORT_COMMAND_FAIL	0xC046	Send or receive cmd fail with VPM.
IMC_ERR_VPM_WAKEUP_SOURCE_NOT_SUPPORT	0xC047	This platform didn't to support this wakeup source.
IMC_ERR_VPM_NO_RESPONSE	0xC048	The VPM response timeout.

6.4 G-SENSOR API ERROR CODE

Code	Value	Description
IMC_ERR_GSENSOR_READ_DATA_ERROR	0xC070	The function succeeds; the platform does not support any APIs.
IMC_ERR_GSENSOR_DATA_INTERNAL_ERROR	0xC071	The function succeeds; the platform supports only primary IIC.

IMC_ERR_GSENSOR_ALARM_MONITOR_ALREADY_ACTIVE_ERROR	0xC072	The function succeeds; the platform supports only SMBus implemented IIC.
IMC_ERR_GSENSOR_DATA_NOT_READY	0xC073	The function succeeds; the platform supports both primary IIC and SMBus IIC.

6.5 DISPLAY API ERROR CODE

Code	Value	Description
IMC_ERR_DISPLAY_FUNCTION_OK_PLATFORM_NOT_SUPPORT_ALL	0xC0D0	The function succeeds; the platform does not support any APIs.
IMC_ERR_DISPLAY_FUNCTION_OK_ONLY_BRIGHTNESS	0xC0D1	The function succeeds; the platform supports only brightness APIs.
IMC_ERR_DISPLAY_FUNCTION_OK_ONLY_SCREEN_CONTROL	0xC0D2	The function succeeds; the platform supports only screen on/off APIs.
IMC_ERR_DISPLAY_FUNCTION_OK_SUPPORT	0xC0D3	The function succeeds; the platform supports all APIs.

6.6 IIC API ERROR CODE

Code	Value	Description
IMC_ERR_IIC_STATUS_OK_PLATFORM_NOT_SUPPORT_ALL	0xC0E0	The function succeeds; the platform does not support any APIs.
IMC_ERR_IIC_STATUS_OK_PLATFORM_ONLY_SUPPORT_PRIMARY	0xC0E1	The function succeeds; the platform supports only primary IIC.
IMC_ERR_IIC_STATUS_OK_PLATFORM_ONLY_SUPPORT_SMBUS	0xC0E2	The function succeeds; the platform supports only SMBus implemented IIC.
IMC_ERR_IIC_STATUS_OK_PLATFORM_SUPPORT	0xC0E3	The function succeeds; the platform supports both primary IIC and SMBus IIC.