DG Technologies 2534 SDK

User Documentation

June 22, 2011

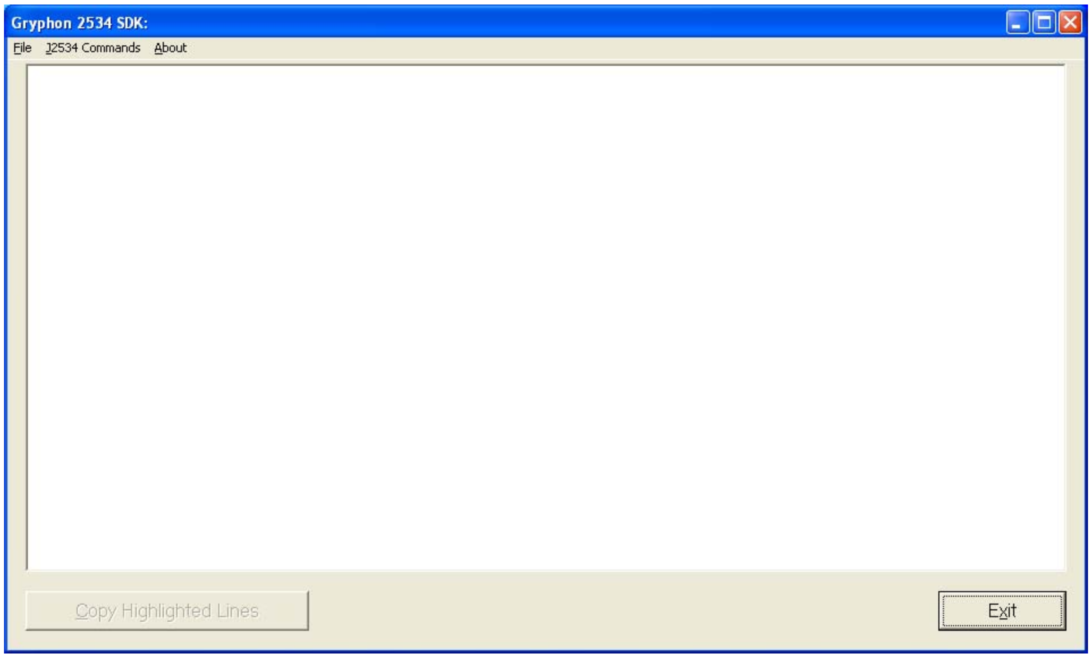
This document describes DG Technologies (DG) SAE J2534 compliant System Development Kit (SDK). The purpose of this application is to facilitate the use of the J2534 API functions using the DG Hardware.

This implementation is based on Version 04.04 of SAE J2534. Of the fourteen (14) functions, only one function is not supported: PassThruSetProgrammingVoltage. This is due to the DG hardware not having this capability at this time.

The protocols supported are: CAN, ISO15765, ISO14230, ISO9141, J1850PWM, and J1850VPW.

Table of Contents

1 Main Display Area

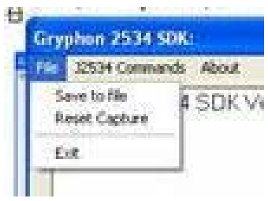


When the “About” Menu Item is selected the Version of the 2534 SDK is displayed as shown below. (This example is for the Gryphon hardware.)

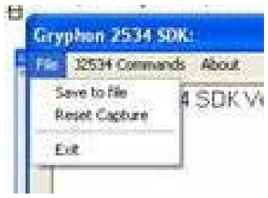
Gryphon 2534 SDK Version 4.04

Note: Select any text in the display area that is highlighted, then the “Copy Highlighted Lines” button will become enabled. If this button is selected, then the text is sent to the clipboard so other applications can use it.

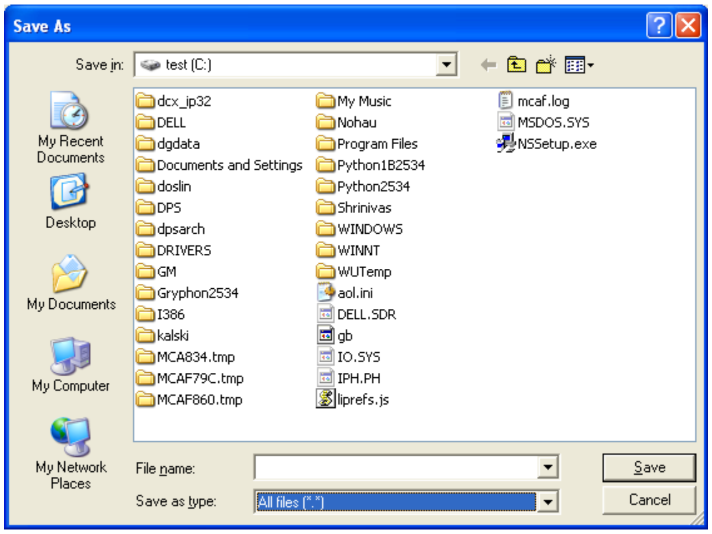
When the “File” Menu Item is selected three selectable menu items are available:



1. Save to File
2. Reset Capture
3. Exit



Selecting "Save to file" will display a standard Windows "Save As" Dialog. This allows the user to save the contents of the Display area to a text file.



Save in: test (C:)

File name:

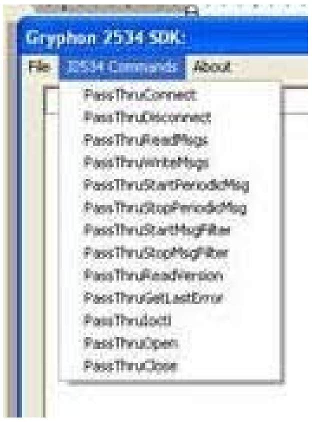
Save as type: All files (*.*)

Selecting "Reset Capture" will allow the user to clear any entries in the display area.

Selecting "Exit" will close the application.

The "J2534 Commands" section when selected displays the following items:

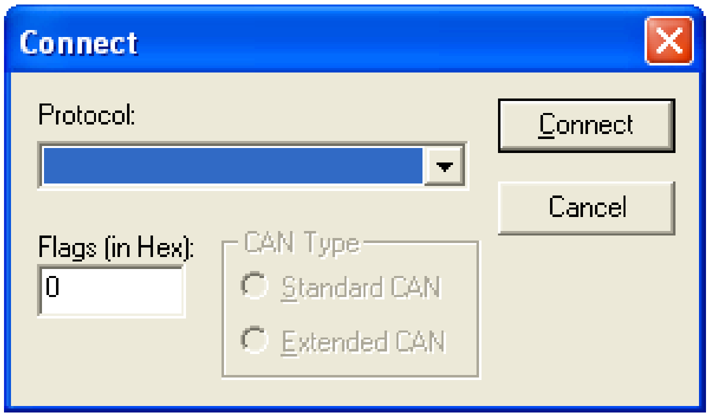
Note: PassThurSetProgrammingVoltage function is not supported. This is due to the Gryphon and Python hardware not having this capacity.



2 J2534 Commands

1. PassThruConnect has a dialog that allows the user to configure which protocol. The following lists the supported protocols that can be configured. The Cancel button closes the dialog.

   PassThruConnect(CAN, 0, &usChannelID);
   PassThruConnect(ISO15765, 0, &ulChannelID);
   PassThruConnect(ISO14230, 0, &ulChannelID);
   PassThruConnect(ISO9141, 0, &ulChannelID);
   PassThruConnect(J1850PWM, 0, &ulChannelID);
   PassThruConnect(J1850VPW, 0, &ulChannelID);



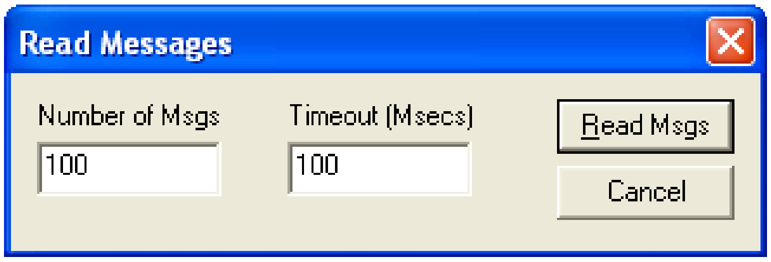
Flags (in Hex): 0

2. PassThruDisconnect ends a connection to a protocol channel. The Cancel button closes the dialog.

   PassThruDisconnect(ulChannelID);

3. PassThruReadMsgs has a dialog that allows the user to configure the number of messages placed in the display area. The number of messages or a Timeout period in milliseconds, whichever occurs first, will end the Read to the display area. After configuring, select the Read Msgs button to display the messages read from the vehicle/ECU. The Cancel button closes the dialog.

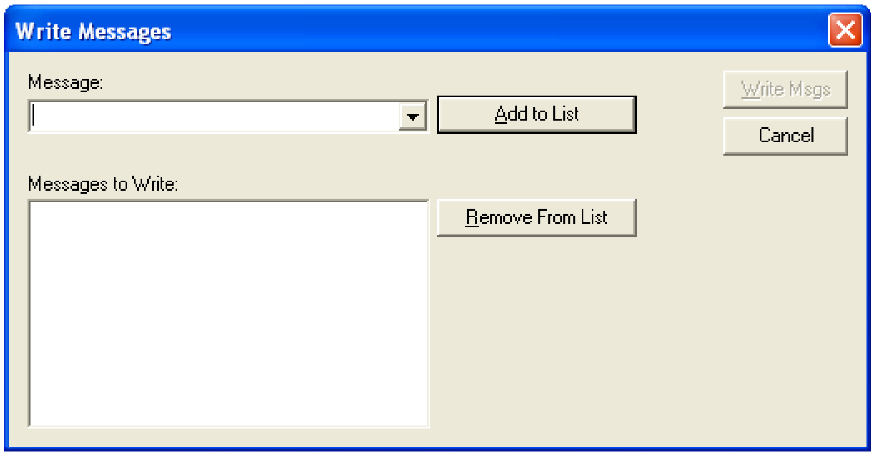
   PassThruReadMsgs(ulChannelID, ptrMsg, &ulNum, 100);



Number of Msgs: 100
Timeout (Msecs): 100

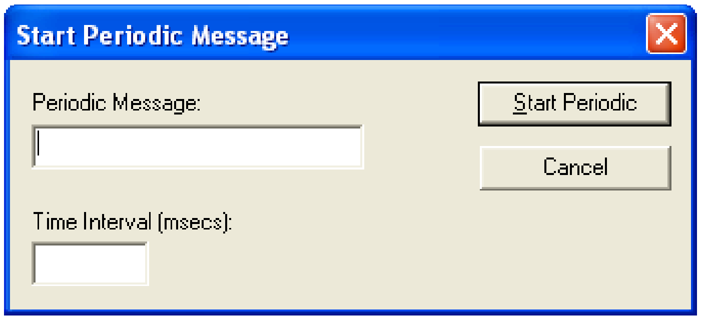
4. PassThruWriteMsgs has a dialog with a Message drop-down box that allows the user to add messages to the “Messages to Write” list. This is done by either directly entering the message, or selecting a previously entered message from the drop-down box. This drop-down box retains the last ten unique messages entered. The list can be edited with the “Remove From List” button. While there are messages in the List the “Write Msgs” button shall be enabled. The messages in the “Messages to Write” list are sent out to the vehicle/ECU when the “Write Msgs” button is selected. The Cancel button closes the dialog.

```
ptrMsg=new PASSTHRU_MSG[ulNumMsgs];
for(unsigned int count=0; count<ulNumMsgs; count++)
{
   memset(ptrMsg+count, 0, sizeof(PASSTHRU_MSG));

   memcpy(ptrMsg[count].Data, ptrTempMsg, size);
   ptrMsg[count].DataSize=ulSize;
   ptrMsg[count].ExtraDataIndex=0;
   ptrMsg[count].ProtocolID=iCurProtocol;
   ptrMsg[count].TxFlags=0x00;
}
PassThruWriteMsgs(ulChannelID, ptrMsg, &ulNumMsgs, 10000);
delete [] ptrMsg;
```



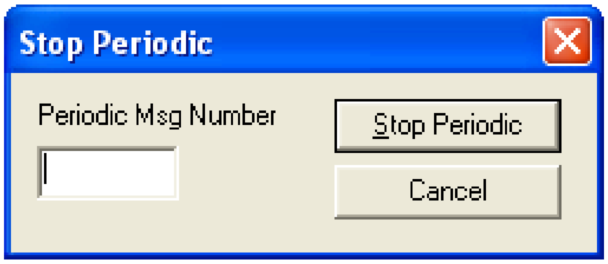
5. PassThruStartPeriodicMsg has a dialog that allows the user to enter the message to be sent out periodically and its time interval. The Cancel button closes the dialog.

```
memset(&ptrMsg, 0, sizeof(PASSTHRU_MSG));
memcpy(ptrMsg.Data, ptrTempMsg, size);
ptrMsg.DataSize=ulSize;
ptrMsg.ExtraDataIndex=0;
ptrMsg.ProtocolID=iCurProtocol;

PassThruStartPeriodicMsg(ulChannelID, &ptrMsg, &ulPeriodicID, ulTimerDelay);
```



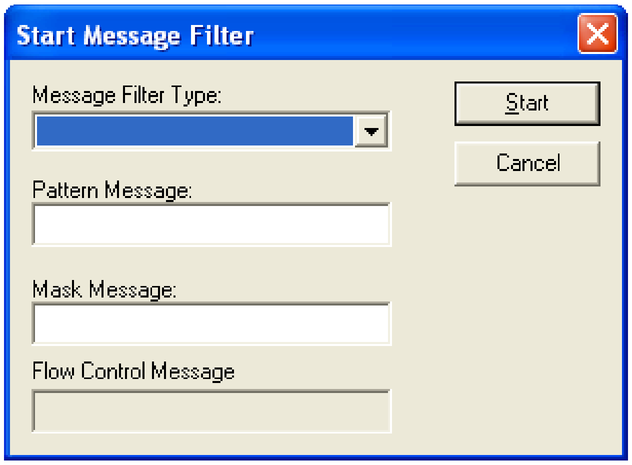
6. PassThruStopPeriodicMsg has a dialog that allows the user to enter the periodic message number, which will identify and stop the periodic message when the “Stop Periodic” button is selected. The Cancel button closes the dialog.

   PassThruStopPeriodicMsg(ulChannelID, 1);



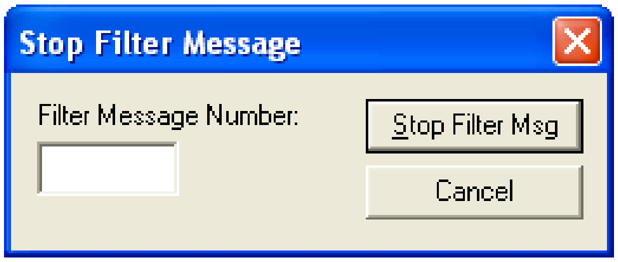
7. PassThruStarMsgFilter has a dialog that allows the user to select a filter type (Pass, Block, or Flow Control), and then enter a Pattern, a Mask, or a Flow Control Message. The Cancel button closes the dialog.

```
memset(&ptrMask, 0, sizeof(PASSTHRU_MSG));
memcpy(ptrMask.Data, ptrTempMsg, size);
ptrMask.DataSize=ulSize;
ptrMask.ExtraDataIndex=0;
ptrMask.ProtocolID=iCurProtocol;

memset(&ptrPatter, 0, sizeof(PASSTHRU_MSG));
memcpy(ptrPattern.Data, ptrTempMsg, size);
ptrPattern.DataSize=ulSize;
ptrPattern.ExtraDataIndex=0;
ptrPattern.ProtocolID=iCurProtocol;

PassThruStartMsgFilter(ulChannelID, PASS_FILTER, ptrMask, ptrPattern, &ulMsgID);
```



8. PassThruStopMsgFilter has a dialog that allows the user to enter the filter message number, which will identify and stop the filter message when the “Stop Filter Msg” button is selected. The Cancel button closes the dialog.

   PassThruStopMsgFilter(ulChannelID, 1);



9. PassThruReadVersion will display the firmware version, DLL version, and the API version in the display area.

```
PassThruReadVersion(ucFirmware, ucDll, ucApi);
Function returned: STATUS_NOERROR
Firmware Version: 20041216
DLL Version: 4.00
API Version: 04.04
```

10. PassThruGetLastError will display the last error, if no error a “Function call successful” comment will be displayed.

```
Last Error: Function call successful
```

   Or

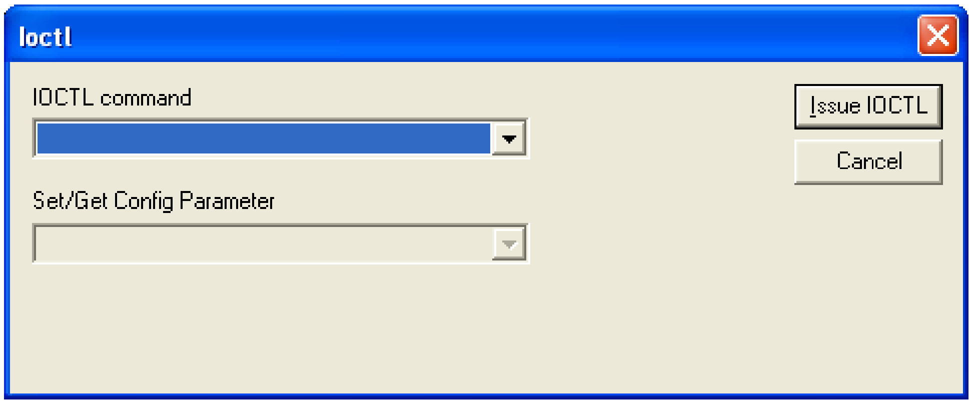
```
Last Error: Function not supported
```

11. PassThruIoctl a dialog that allows the user to select from a list of IO controls (IOCTLS), and Set/Get their configuration parameter. The Cancel button closes the dialog.

```
ConfigList.NumOfParams=1;
ConfigList.ConfigPtr=&Config;
Config.Parameter=1;
PassThruIoctl(ulChannelID, GET_CONFIG, &ConfigList, NULL);
```

Or

```
ConfigList.NumOfParams=1;
ConfigList.ConfigPtr=&Config;
Config.Parameter=3;
Config.Value=0;
PassThruIoctl(ulChannelID, SET_CONFIG, &ConfigList, NULL);
```



12. PassThruOpen is the initial function to be issued, which opens the link to the hardware (Gryphon, Python or VSI 2534).

    PassThruOpen();

13. PassThruClose ends the link to the hardware

    PassThruClose();

3 Known Issues:

1. The Read Messages dialog shows the Timeout as Msecs, it should be msecs – milliseconds
2. The function PassThruStartMsgFilter is shown in the display area as PassThruStartFilter
3. The function GetLastError is not shown in the display area, only the return value.