

Westpac Technology

Tomorrow's CAN Technology for you Today!

WT ISO 14229 UDS CAN



User Manual

V2.00

3 Aug 2020

Diagnostics



Scripting



Bootloader



Quick Start Guide

Preliminary

Table of Contents

1	Preface	3
2	Introduction	3
3	System Requirement and Installation	3
3.1.	System Requirement	3
3.2.	Operating System Requirement	3
3.3.	Installation	3
4	Start-up “WT ISO14229 UDS CAN”	4
4.1.	Loading the Initialisation (Ini) file	4
4.2.	CAN Device Hardware Setting	4
4.2.1.	USB & PCI type devices	4
4.3.	Connecting and disconnecting the CAN BUS	5
5	The Main Panel Screen	5
5.1.	Title Bar	6
5.2.	Menu Bar	6
5.2.1.	File	6
5.2.2.	Hardware Setting	6
5.2.3.	CAN BUS On/Off	6
5.2.4.	Extra Features: Scripting & Bootloader	6
5.2.5.	Help	6
5.3.	Transmit & Receive ID setting	7
5.4.	Diagnostic Session selection	7
5.5.	Information Display Panel	7
5.5.1.	Display Panel control buttons	7
5.6.	UDS Commands & Controls	7
5.6.1.	Transmit Command String	8
5.7.	Panel for additional control options	9
5.8.	Bootloader Control Panels	11
5.8.1.	Individual button operational controls	11
5.8.2.	Input and editing of the Finger Print data	14
5.8.3.	Editing of the automated Bootloading operational sequence	16
5.8.4.	Create/Edit the Download Command operation	16
5.8.5.	Managing .fcs file	17
5.8.6.	Execute the “Auto Flash Sequence”	18
5.9.	CAN device information	19
5.10.	DTC status Commands and Controls	19
5.11.	“Tester Present” signal Controls	19
5.12.	Security Access Controls	19
5.13.	Password access- Supplier Specific	20
5.14.	ECU Reset buttons	20
7	Appendix A: A quick start guide for preparation to Bootloader Operation	21
8	Appendix B: Example of Re-flash Download Operation	22
9	Appendix C: Sample of setting in INI File for Bootloader controls	23

This document is a property of Westpac Technology, Westpac Technology reserved the rights to change the contents of this document without prior notice, for further information, please contact us at www.westpactech.com

1 Preface

This document briefly describes the command screen of Westpac's WT ISO14229 UDS CAN (Former called WT ISO14229 UDS CAN 2018) including Scripting commands and Boot-Loader functions.

2 Introduction

The WT ISO14229 UDS CAN initialises the operational parameters/settings via the imported Initialisation (ini) file; this initialisation file permits user to customise the communication protocols as required by specific project.

The Scripting feature allows engineers to write quick and simple programming procedures for testing the ECU performance data, the Script Test operation can be repeated many times, stop on specific condition and log data into a file for post analysis.

The Bootloader function demonstrated in this document shows a particular Customised operation sequence to re-flashing the ECU program codes.

3 System Requirement and Installation

3.1. System Requirement

- ☐ CPU Speed: Pentium 1.6GHz minimum or equivalent or higher.
- ☐ Memory: 512MB Memory or above
- ☐ Connectivity: USB Port or PCI Slot or PCI Express Slot on the PC for Kvaser CAN Product Range

All Kvaser CAN Device are required to have firmware version 1.9 or above and Kvaser System Driver for Windows V5.12.0 or above; which can be downloaded from <http://www.kvaser.com/downloads/>. Also please ensure "Microsoft Visual C++ 2010 x86 Redistributable" is installed; following link provides more details: <https://www.microsoft.com/en-au/download/details.aspx?id=1639>

Note: For Laptop user, please ensure the laptop is main power and not using battery power, this is because when Laptop is using battery power its operating speed will be greatly reduced and the performance will not at its best.

3.2. Operating System Requirement

The CAN3 Simulator supports the followings Operating System:

- ☐ Windows 7 64/32-bits
- ☐ Windows 8 64/32-bits
- ☐ Windows 10

CAN3 Simulator requires Microsoft .NET Framework Version 3.5 be installed. For more information about this, you can visit <http://www.microsoft.com/downloads/en/default.aspx> Search for the key words **.NET Framework 3.5**. This .Net Framework 3.5 covers .Net Framework 2.0 & 3.0.


3.3. Installation

To install **WT ISO14229 UDS CAN** simply clicks **Setup.exe** and follows the instruction given on the screen.

The **WT ISO14229 UDS CAN** will be installed into the **Start Menu** under the **WT** folder.

4 Start-up “WT ISO14229 UDS CAN”

You can start this “WT ISO14229 UDS CAN” by clicking the following commands:

→  → “All Programs” to activate this utility service as shown in

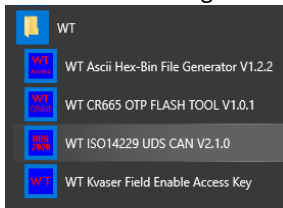


Fig 1.

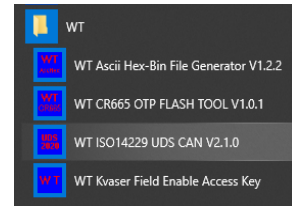


Fig 1: Activation steps for “WT ISO14229 UDS CAN”

4.1. Loading the Initialisation (Ini) file

Upon activation of “WT ISO14229 UDS CAN”, a Window will pop up for selecting a Database file for Initialisation, this Ini file contains all the parameters that are available specific for this UDS application. In this example, a file name “UDS Demo ECU CAN-Generic 180805.ini” is provided for use the ECU demo Unit- See Fig 2 below.

After loading the INI file, you can always change the Ini file for other projects by clicking “File” → “Open INIT File” to select other INI file as shown in Fig 3

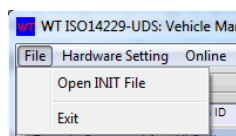


Fig 3: Steps to select other

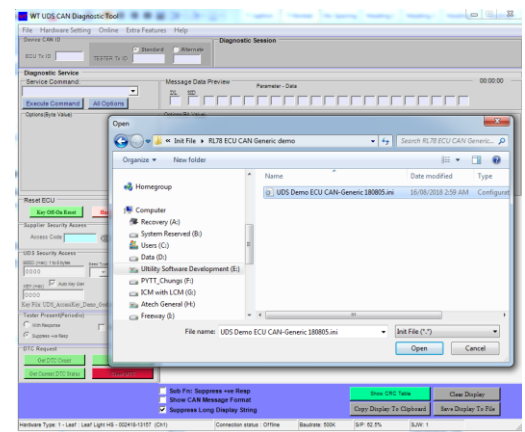


Fig 2: Example screen for loading the initialisation Database file

4.2. CAN Device Hardware Setting

You need to select & set the Kvaser CAN device in accordance to the required parameters, click “Hardware setting” to open the screen for CAN device Hardware Configuration for the following.

- ☐ Choose available CAN Device (See section 4.2.1 below)
- ☐ Baud rate
- ☐ Sampling Point
- ☐ SJW (Synchronise Jump Width)

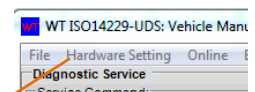
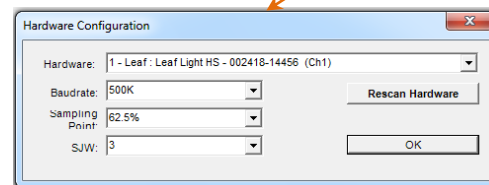


Fig 4: Steps to set the CAN device for CAN communication



4.2.1. USB & PCI type devices

There are basically two types of connecting devices from Kvaser, namely the USB and PCI. The USB device would have the Key enabled by the supplier upon purchased and is transparent to the user; whilst the PCI would require user to manually import the Access key that is provided by the supplier. Following example shows the procedures for importing the Access Key for PCI devices:

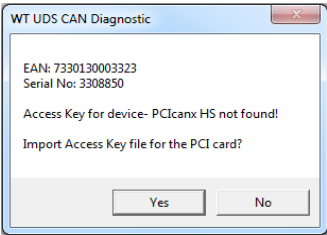


Fig 5: Example of prompt message for missing Access Key

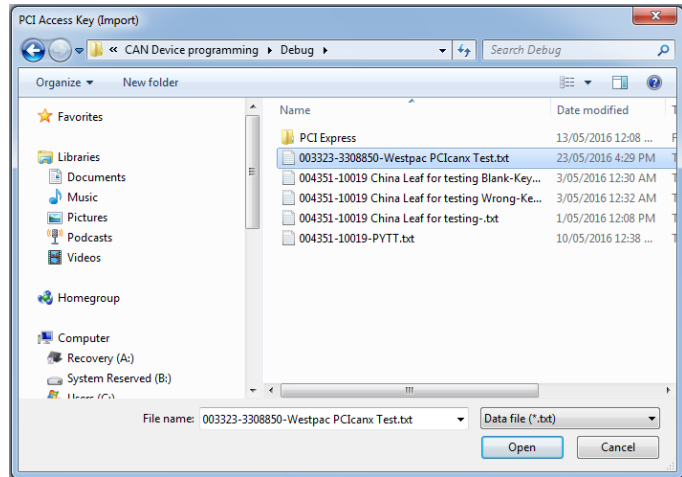


Fig 6: Example screen for entering the Supplier Provided Access Key for the PCI interface card

After clicking the “Yes” button, a new screen (Fig 6) will pop up for you to provide the location of the Access Key file that was provided by the supplier. After the Access Key file has been selected, click “Open” button to complete the operation. Please note that once the valid Access Key file has been provided, it will not ask for it again.

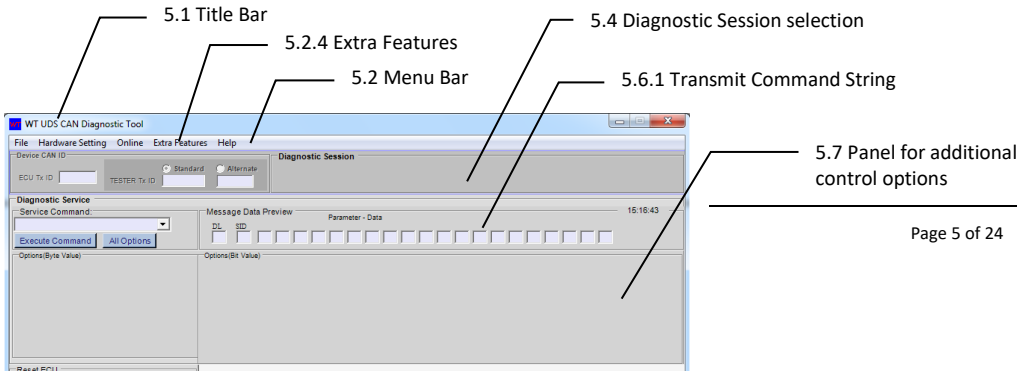
4.3. Connecting and disconnecting the CAN BUS

When the INI file is loaded and the CAN Device Hardware settings are done, Click “Online” button to connect the CAN Device Hardware on to the CAN BUS. The “WT ISO14229 UDS CAN” is now ready to send and receive CAN messages on to the CAN BUS.

After clicking the “Online” button, it will change to “Offline”, click this “Offline” to disconnect the CAN device from the CAN BUS.

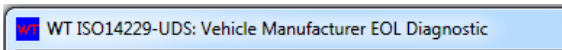
5 The Main Panel Screen

The display screen panel as shown below shows the control available, for ease of explanation, the discussion of the usage will be divided into sections as follow:



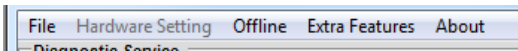


5.1. Title Bar



This title bar illustrates the descriptions of the project undertaking; the description text string is stored in the loaded INI file.

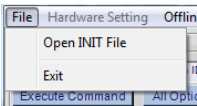
5.2. Menu Bar



There are 5 options available on the Menu Bar, they are as follows:

5.2.1. File

Picture on the right hand side shows the display dialog after "File" option has been clicked.



Two options are available; they are

- Open INIT File : To load an INI file data base containing all the UDS commands and information for a particular project
- EXIT : To exit this UDS utility program

5.2.2. Hardware Setting

See section 4.2 CAN Device Hardware Setting

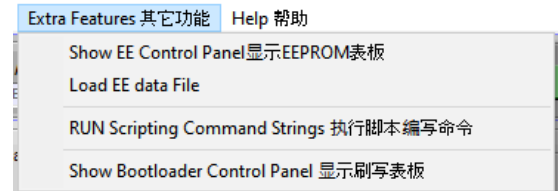
5.2.3. CAN BUS On/Off

This option command is to toggle the connection between the Kvaser CAN device and the CAN BUS.

When it is displaying "Online" it means it is ready to engage the CAN BUS, while when it is displaying "Offline"- the CAN BUS will be disengaged.

5.2.4. Extra Features: Scripting & Bootloader

When this “Extra Feature” is selected; there are 5 available options as shown in the picture on the right-hand side. The enable/disable of the available feature is set in the INI file.



- <Show EE Control Panel>: This is Customer specific function, please contact us for details
- <Load EE data File>: This is Customer specific function, please contact us for details.
- <RUN Scripting Command Strings>: this Scripting features provide some basic programming sequence to make the testing easier and repeatable. There also have scripting commands to save the received diagnostic data into a file for post analysis or into Excel Worksheets for graphical illustrations. For full details, please refer to separate User Manual “UM WT ISO14229 UDS CAN Scripting”.
- <Show the Boot-loader Control Panel>: Show the control panel for the Boot-loader functions- See section 5.8. for more details.

5.2.5. Help

There are two options for you to choose, namely the EULA (End User Licence Agreement) and About display of the useful information about this software tool:

5.2.5.1. UCLA

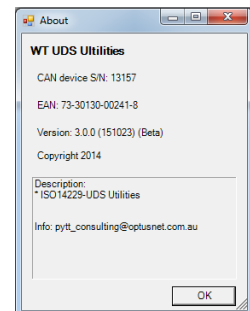
To show the terms and condition of use for this Software.

5.2.5.2. About

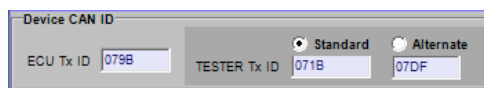
The picture as shown on the right hand side shows the example of the “About” information Window after clicking the “About” button.

The display illustrates some useful information as follows:

- About the connected Kvaser device
- About the Version of this Utility program
- Contact details for further information



5.3. Transmit & Receive ID setting



The above picture shows the CAN device ID as follows:

- ECU Tx ID: The ID that the ECU will respond
- TESTER Tx ID- Standard: This is the ECU specific CAN ID that the Tester is send to the designated ECU
- TESTER TxID- Alternate: This CAN ID is the Tester sends out as Broadcast mode for all ECU to receive

These CAN IDs are set in the Initialisation file.

5.4. Diagnostic Session selection



The above picture shows the four diagnostic session button available for selection, they are as Follow:

- <DEFAULT> : Default session- Command 10h, 01h
- <PROGRAMMING>: Programming session - Command 10h, 02h

- <EXTENDED> Extended session - Command 10h, 03h
- <SUPPLIER>: Supplier session - Command 10h, 61h

5.5. Information Display Panel

This display panel shows the activities and dialogue of operation; designed for User to trace and verify the interactive sequences.

5.5.1. Display Panel control buttons

There are 3 buttons available for control the Display Panel; they are as follows:

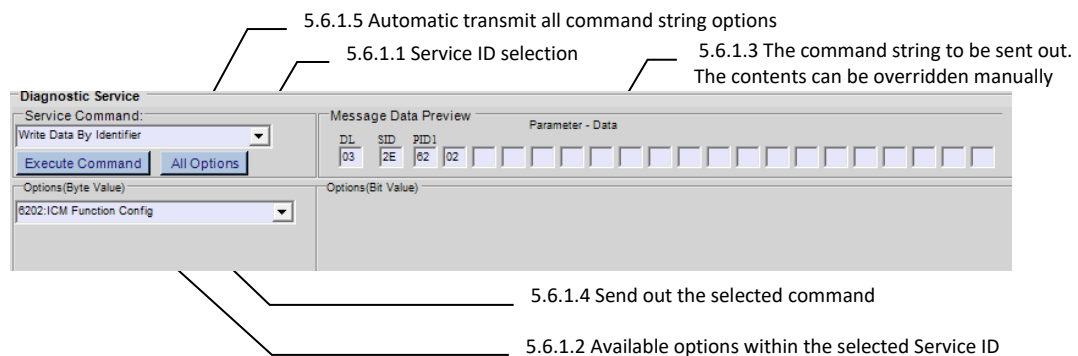
- <Clear Display>: Clear the Display screen panel
- <Copy Display To Clipboard>: Copy the screen contents to Clipboard
- <Copy Display To File>: Copy the screen contents to a file

5.6. UDS Commands & Controls

This section discusses the control and services available for sending out the UDS commands string.

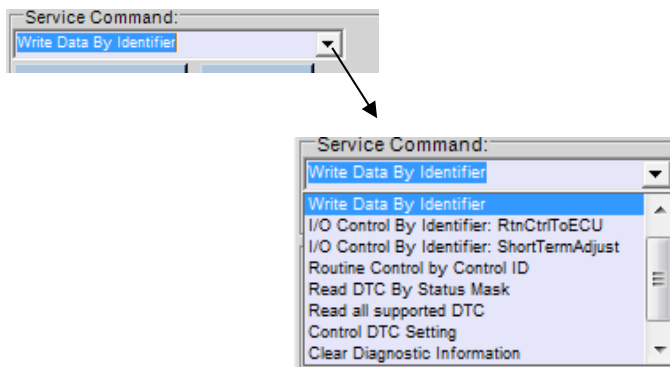
5.6.1. Transmit Command String

Following diagram shows the control available to select a command string to be sent out with a click on the “Execute Command” button. The command options are stored in the Ini file as described in section 5.2.1.



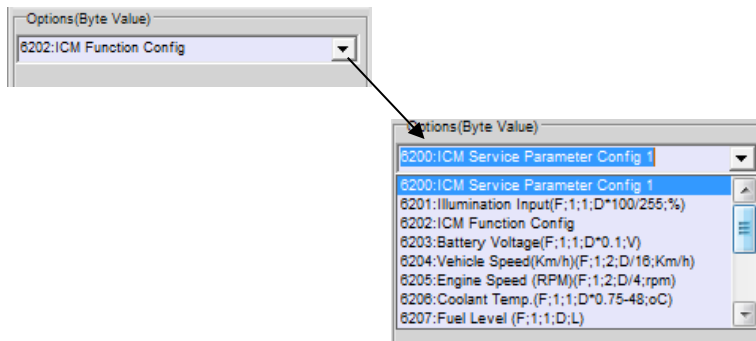
5.6.1.1. Service ID selection

When selection button is clicked as per the pictures shown below, a drop down menu will be displayed with the available Service ID for selection.



5.6.1.2. Option (Byte Value)

When the selection button is clicked as per the pictures shown below, a drop down menu will be displayed with the available option data bytes i.e. DID for selection



5.6.1.3. Command String

Below picture shows the constructed Command String from the selection of (A) and (B) above. From this example, the constructed String is-

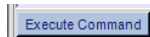
- DL (Data Length) : 3 bytes
- SID (Service ID) : 0x22 Read Data By Identifier
- PID (or DID) : 0x6201



The value in each cell can be changed manually should it be required.

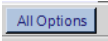
5.6.1.4. Send out the selected command

When the command string is ready as shown in the “Message Data Preview” panel, click “Execute Command” button to send out the diagnostic string.



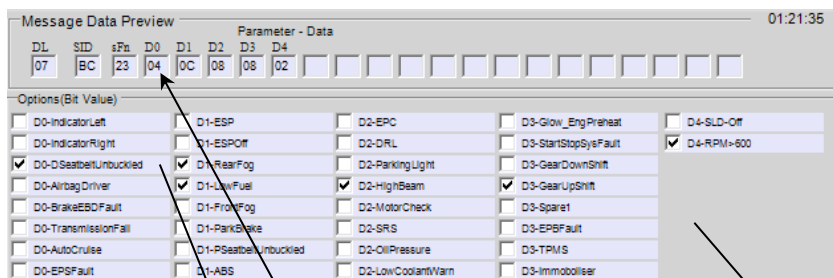
5.6.1.5. Automatic transmit all command string options

This feature is useful for sending out dedicated test sequence.

When this “All Options” button  is clicked, all the available options within the selected Service ID will automatically be transmitted in sequential manner.

5.7. Panel for additional control options

Below picture shows the additional control options for bit controls; that is in a 8 bit data byte, it allows for individual bit function control, the send byte will automatically be updated with the bits selection- in the example shown below, b2 is checked and the value 0x04 is set.



Bit 2 is selected and value 0x04 is updated

Panel of Additional control option

To enable this bit option, set the data option to bit as shown in the example below:

```
//*****
%Service=ICM-FnCtrlByID: Warning Light-Digital Out
%SID=BC
%option1=01,IndicatorLeft,Bit
%option1=02,IndicatorRight,Bit
%option1=04,DSeatbeltUnbuckled,Bit
%option1=08,AirbagDriver,Bit
%option1=10,BrakeEBDFault,Bit
%option1=20,TransmissionFail,Bit
%option1=40,AutoCruise,Bit
%option1=80,EPSError,Bit

%option2=01,ESP,Bit
%option2=02,ESPOff,Bit
%option2=04,RearFog,Bit
%option2=08,LowFuel,Bit
%option2=10,FrontFog,Bit
%option2=20,ParkBrake,Bit
%option2=40,PSeatbeltUnbuckled,Bit
%option2=80,ABS,Bit

%option3=01,EPC,Bit
%option3=02,DRL,Bit
%option3=04,ParkingLight,Bit
%option3=08,HighBeam,Bit
%option3=10,MotorCheck,Bit
%option3=20,SRS,Bit
%option3=40,OilPressure,Bit
%option3=80,LowCoolantWarn,Bit

%option4=01,Glow_EngPreheat,Bit
%option4=02,StartStopSysFault,Bit
%option4=04,GearDownShift,Bit
%option4=08,GearUpShift,Bit
%option4=10,Spare1,Bit
%option4=20,EPBFault,Bit
%option4=40,TPMS,Bit
%option4=80,Immoboliser,Bit

%option5=01,SLD-Off,Bit
%option5=02,RPM>600,Bit

%Descriptor=sFn,D0,D1,D2,D3,D4
%Parameter=23,%option1,%option2,%option3,%option4,%option5
%End
```

5.8. Bootloader Control Panels

..... Cont from section 5.2.4 Extra Features

After clicking “Show Bootloader Control Panel” option, it will activate multiple display panels to help you to input and control your re-flashing operation; the panels are as follows:

- Individual button operational controls
- Input and editing of the Finger Print data (i.e. Tester and operator signature, as well as the Date code)
- Editing of the automated Bootloading operational sequence

5.8.1. Individual button operational controls

Fig 7 shows the available buttons to initiate the Bootloader command, this individual command buttons is designed to aid in the ECU development and testing of the individual thereof.

This Bootloader Control Panel can be set in the INI file for initial Start-Up, the command to be in the INI file is as follows:

%ShowBootloaderPanel=True

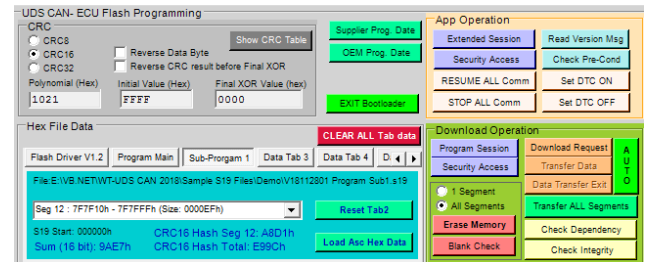
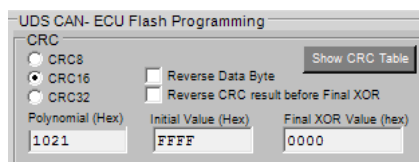


Fig 7: Individual button control for Bootloader operation

This Bootloader function has the following features:

- Option for CRC8, CRC16 and CRC32 Checksums calculation
- Up to 6 control tabs for Multiply Hex Code downloads
- Commands for Application control and communication
- Command for Bootloader control and communication
- Set Finger Print identification data i.e. Operator ID, Programming date, etc

5.8.1.1. CRC -8, CRC-16 and CRC-32 settings



The above picture shows the panel for CRC selection and settings, it allows for full CRC operational setting as shown.

The default setting for

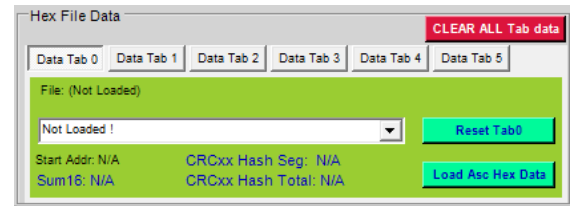
- CRC8 is CCITT-8 with Polynomial 07h;
- CRC16 is CCITT-16 with Polynomial 1021h; and
- CRC32 is the commonly used Polynomial 04C11DB7h;

The setting for the CRC can be set in the INI file as follows:

```
//*****
//**** CRC Masks
//*****
// CRC-Type, Polynomial(h), Initial Value(h), Final XOR Value(h), Reverse Data Byte(Boolean), Reverse CRC Result Before Final Xor(Boolean)
%CRC=8,07,00,00,False,False // CCITT-8
%CRC=16,1021,FFFF,0000,False,False // CRC-CCITT (16 bit)
%CRC=32,04C11DB7,FFFFFFFF,FFFFFFFF,True,True // CCITT-32
%CRCStartUp=CCITT16 // Select CRC16 as Start-up CRC
```

5.8.1.2. Load Asc Hex Data in File

The Right Hand Side picture shows the control panel for loading the Hexadecimal data in files, the available control buttons are as follows:



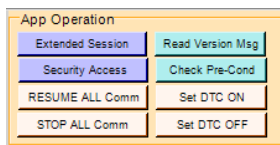
- <CLEAR ALL Tab Data>: this button is to completely reset/clear all the Data Tab contents
- <Reset Tabx>: reset the contents of the current Tabx; where x = 0 to 5
- <Load Asc Hex Data>: Import either Intel Hex or S19 formatted data file as in one program or segments in to the selected Data Tab, you can import up to 16 segments into each Data Tab. The CRC per each segment will be calculated, so as well as the CRC for the total segments will also be calculated. This combined CRC is for one Hex data file that has been divided into multiple segments; the combined CRC can be used in the "Check Integrity" inquiry command. In addition to the CRC, an optional 16 bit Sum Checksums will also be calculated, this 16 bit Sum Checksums can be used by the program itself for start up self check.
- <Data Tab 0 to Tab 5>: All Data Tab have the same control and features, you can use one Data Tab per particular program code, i.e. One Data Tab for embedded Flash Routine to be run in RAM, one for Program code A, one for Program code B, one for Data A, one for Data B, etc. and Data Tab can load up to 16 segments of code/data.

Please note that the addressing for the Tab supports up to 32 bits addressing with actual memory linear size of 0x1000000 bytes with upper 8 bits as Pages, that is for 32 bit address, it has 256 pages with each page of buffer space from 0x000000 to 0xFFFFF

Example:

- Address 0x0000FFFF: Page 0x00 with Data address at 0x00FFFF
- Address 0x11A01234: Page 0x11 with Data address at 0xA01234

5.8.1.3. Application operation



The above picture shows the control that is to interact with the ECU application section, the control buttons are as follows:

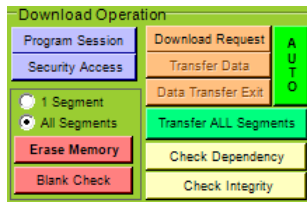
- <Read Version Msg>: Read the Version message of the from the ECU to determine it is a valid ECU for re-flashing
- <Check Pre-Condition>: Command button sends out the command to inquire the ECU whether the ECU is in a good condition and ready for re-programming. Example of the Pre-Conditions are:
 - Valid Voltage supply i.e. Between 9V to 16V;
 - Engine not running;
 - Vehicle is stationery;
 - Etc

Normally, when all the Pre-Conditions are met, the ECU will return a Positive Response code before jump directly in to the Bootloader section; the Application routines will cease from operation. Set DTC ON: Use this command to broadcast to instruct all ECUs to resume logging of DTC

- <Set DTC OFF>: Use this command to broadcast to instruct all ECUs to stop logging of DTC
- <RESUME All Comm>: Use this command to broadcast to all ECUs to resume CAN communication; except diagnostic commands
- <STOP All Comm>: Use this command to broadcast to all ECUs to Stop CAN communication; except diagnostic commands
- <Extended Session>: Send command code 10h, 03h to command the ECU into Extended session
- <Security Access>: This button for Seed type and Key is configurable in the INI file. When it is clicked, it will send out the request for Seeds; upon received of the Seed, it will automatically called the DLL file to

calculate the access Key then send it out to the ECU. Please make sure the “Auto Key Gen” in the UDS Security Access Panel is checked.

5.8.1.4. Download operation



The above picture shows the control buttons for interaction with the ECU download operation, the control buttons are as follows:

- <Programming Session>: Send command code 10h, 02h to command the ECU into Programming session
- <Security Access>: This button for Seed type and Key is configurable in the INI file. When it is clicked, it will send out the request for Seeds; upon received of the Seed, it will automatically call the DLL file to calculate the access Key then send it out to the ECU. Please make sure the “Auto Key Gen” in the UDS Security Access Panel is checked.
- <Blank Check>: This button is to provide user to check the intended memory section is blank and ready for Flashing; this feature is not included in the UDS, we include here as a good option for user to inquire the Memory status i.e. ensure it is Blank before the Flashing can commence.
- <Erase Memory>: This command is to erase the section of memory as per the selected segment in the Data Tab.
- <Download Request>: This command to initiate the Data Transfer sequence of the selected segment, in the command message it contains the Start address and Data size; the counting of Data size from 0 or 1 can be set in the INI file.
- <Transfer Data>: Transfer the segment data in block or multi-blocks with the size limit provided in the response data of the <Download Request>
- <Data Transfer Exit>: This command to instruct the Bootloader to stop the Data Transfer routine. The ECU returns a positive response with a predefined code if any; this predefined code can be a 8 bit sum checksums for the Tester to verify and can be set in the INI file as follows:

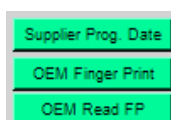
%TransferExit8bitSumCSumInvert=On // This is the 8 bit Inverted Checksums for Data Block Transfer

- <AUTO>: This Button is included to automate the sequence of <Download Request> <Transfer Data> and <Data Transfer Exit> for the selected segment. This AUTO feature would also to overcome some ECUs timing limitation between the Download-Transfer sequences; in which the manual operation for the 3 download sequences would exceed the operational time window.
- <Transfer ALL Segments>: This Button is included to automate the transfer of all segments in the selected Data Tab.
- <Check Dependency>: This command is to inquire the ECU to verify the last Data block is correct or not; normally the verification process is a CRC8, CRC16 or CRC32.
- <Check Integrity>: This command is to inquire the ECU whether the downloaded program (whether it is one single download or in multiple segments) that has been re-flashed properly and ready for ECU reset.

Note: For whatever the reasons, some OEM has taken the counting of memory size to be started from 1 instead of 0. Although it is not clearly specified, it is understood the counting is to start from 0, that is for 64K (FFFFh) bytes, it only requires 16 bits, but if it is counting from 1, it requires 17 bits e.g. 10000h. Following command can be set in the INI file to choose between the counting from 0 or 1:

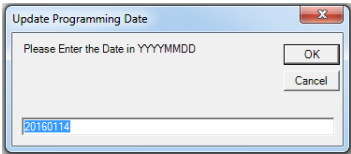
%BLDataSizeCountFrom1=True // Bootloader function- Memory size count from 0 or 1; True =1, False= 0

5.8.1.5. Programming dates - Finger-Print records



The picture as shown above shows the 3 control buttons for Programming and readout of the date of programming; this programming date with other information serves as the Finger Print of the Re-Flashing operation.

- <Prog. Supplier Date>: Reserved for use by the Supplier to program into the non-volatile memory the date and other information i.e. Tester ID or Operator ID. Clicking this button will have a message box to pop up (an example is shown on the right) for entering the Date code.
- <OEM Finger Print>: Strictly be used by OEM operator to record the date and other information (Finger Print data) in to the non-volatile memory i.e. Tester ID or Operator ID. See section 5.8.2 below for details.
- <OEM Read FP>: This command button in to inquire from the ECU, all the registered Finger Print details from the Non-Volatile memory.



5.8.2. Input and editing of the Finger Print data

For each flashing of a program, it requires some sort of associated data be recorded for traceability, these data can be Operator ID, Test Equipment ID, Date of the operation, etc these data have a jargon term as Finger Print.

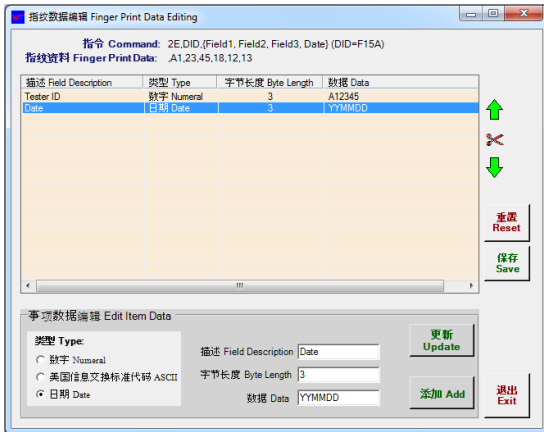
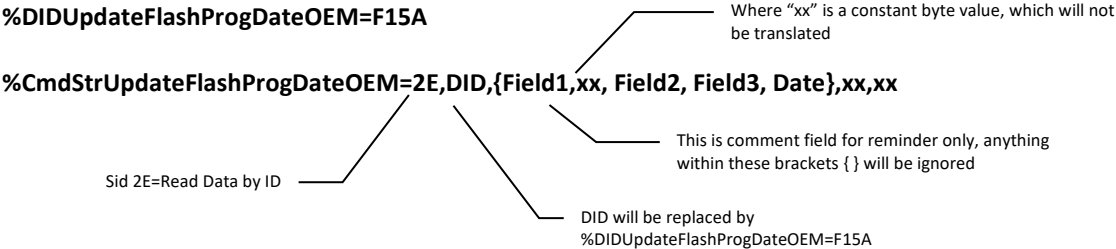


Fig 8: Editing Window for Finger Print Data

When this command “%CmdStrUpdateFlashProgDateOEM” is set in the INI file, an editing Window (Fig 8.) will be popped up for you to enter the required Finger Print Data. Otherwise if this command “%CmdStrUpdateFlashProgDateOEM” is not in the INI file, the Window of Fig 8. Will not be shown.

In this example, the command in the INI file is as follows:



When the curly bracket ‘{ }’ is used, only the variable field within the curly bracket will be translated, otherwise; all the variable field within the entire command string will be translated.

Example A: Using the Curly bracket { }

- command string in INI file is %CmdStrUpdateFlashProgDateOEM=2E,DID,{ProgDate,TesterID},00,00;
- ProgDate = YYYYMMDD; and
- TesterID = 123456
- Fixed data = 00,00

Below screen shot that use curly bracket shows the Command and the Finger Print data, which when it is completed, the command string will be 2E,F1,5A,20,08,03,12,34,56,00,00

指纹数据编辑 Finger Print Data Editing

指令 Command: 2E,DID,(ProgDate,TesterID),00,00 (DID=F15A)

指纹资料 FPD Data: (20,08,03,12,34,56)

输入的指纹数据
Finger Printer Data

描述 Field Description	类型 Type	字节长度 Byte Length	数据 Data
TesterID	十六进制数 Hex Numeral	3	123456
ProgDate	日期 Date	3	YYMMDD

Example B: Without the Curly brackets { }

- command string in INI file is %CmdStrUpdateFlashProgDateOEM=2E,DID,ProgDate,TesterID,AB,CD;
- ProgDate = YYMMDD; and
- TesterID = AB1234
- Fixed Data = E1 and F1

Below screen shot that use curly bracket shows the Command and the Finger Print data, which when it is completed, the command string will be 2E,F1,5A,20,08,03,AB,12,34,E1,F2

指纹数据编辑 Finger Print Data Editing

指令 Command: 2E,DID,ProgDate,TesterID,E1,F2 (DID=F15A)

全指纹命令 Full Cmd: 2E,DID,20,08,03,AB,12,34,E1,F2

全指纹命令
Full Command String

描述 Field Description	类型 Type	字节长度 Byte Length	数据 Data
TesterID	十六进制数 Hex Numeral	3	AB1234
ProgDate	日期 Date	3	YYMMDD

You can use the **Edit Item Data** below to enter the Finger Print Data in the required order sequence

指纹数据类型选择:

- 十六进制数 Hex Numeral: 2 位BCD/HEX
- ASCII: 美国信息交换标准代码
- 日期 Date: YYMMDD, DDDMMYY, etc

事项数据编辑 Edit Item Data

类型 Type:

☒ 十六进制数 Hex Numeral

☐ 美国信息交换标准代码 ASCII

☐ 日期 Date

描述 Field Description: TesterID

字节长度 Byte Length: 3

数据 Data: AB1234

更新 Update

添加 Add

更新选择中的
指纹数据

添加这个指纹
数据

指纹数据: 描述,
字节长度及数据

Steps to add an item data

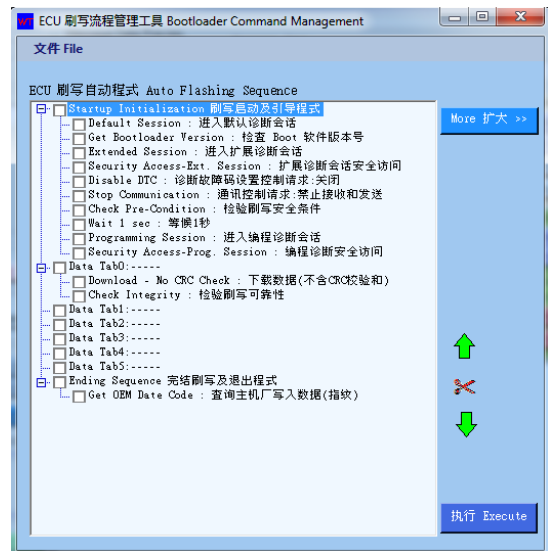
- Step 1: Select “类型 Type” i.e. “十六进制数 Hex Numeral”, “美国信息交换标准代码 ASCII” or “日期 Date”
- Step 2: Enter the description in “描述 Field Description”
- Step 3: Enter the byte length of the data in “字节长度 Byte Length”
- Step 4: Enter the data in accordance to the “类型 Type”
- Step 5: Click the button “添加 Add” to add this data as new or “更新 Update” to update the current selected Finger Print Item

5.8.3. Editing of the automated Bootloading operational sequence

This also refers to as “One Button Auto Flashing”; in which it automatically completes the Data Transfer or flashing of all segments in the order of the pre-defined command sequences.

When “Bootloader Control Panel” as described in section 5.8 is shown, the following Window “ECU 刷写流程管理工具 Bootloader Command Management” will also be shown:

This Window is to allow you to edit the command sequence as required by the ECU to Flashing:



5.8.3.1. Tree View Branches

In the View Tree Structure, it shows the 8 branches to set the download operation commands.

5.8.3.1.1. Branch 1: Start up initialisation

This Startup- Initialisation branch is to hold the sequence of commands as required by the ECU to get ready to receive the transfer/flashing of the Data in the 6 of the Data Tabs.

5.8.3.1.2. Data Tab0 to Data Tab5

These are the 6 Data Tabs available for loading 6 different programs to be transferred/Download. Please load the Programs into the Data Tabs in the order of Program priority sequence.

For example, if the ECU requires a Flash Driver to be downloaded 1st before the flashing of the program or Data, the Flash Driver program detail shall be imported to Data Tab 0 and the program to be re-flashed shall be imported to Data Tab 1 and other sequence of program thereto to other Data Tabs in order of the sequence.

5.8.3.1.3. Ending Sequence

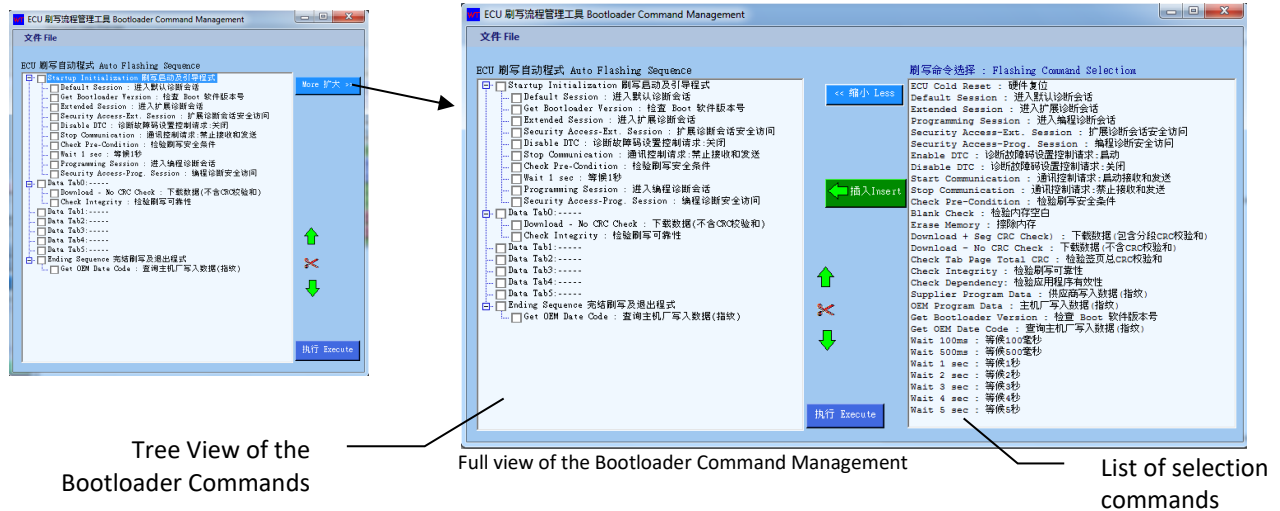
This branch holds the command sequence as required to verify the Download before exit, these exit commands can be:

- Program the Date code as used as Finger Print of the operation
- Check the Program Integrity i.e. is the program has been successfully re-flashed
- ECU Reset
- Etc.

5.8.4. Create/Edit the Download Command operation


The Download command sequence varies from ECU to ECU, thus the command sequence is required to be arranged and set in the required orders, following will discuss the available tool to edit the required command sequence.

The available commands is hidden from the Window Panel, but you can have “Full view of the Bootloader Command Management” by clicking the this “More” button More 扩大 >>, the diagram below shows the full controls of the editing management.






5.8.4.1. Add a command to the Bootloader command Branch

In the Full view of the Bootloader Command Management Window, it shows the selection list of available commands on the right hand side. To add a command to the branch of the view tree, simple follows the following steps:

- Step 1: Select the Branch or location of the command as the location to be inserted
- Step 2: Select from the Command List
- Step 3: Press this Insert button  to insert the selected Command from the Command list to the designated Branch of the Bootloader Commands

5.8.4.2. Managing the Commands in the Bootloader Command Tree

There are 3 available controls available for managing Command in the Bootloader Command Tree; namely, Move-Up, Move-Down and Delete.

- Press this Up Button  to move the selected Command upward
- Press this Down Button  to move the selected Command downward
- Press this Cut/Delete button  to delete the selected Command or Clear the selected branch

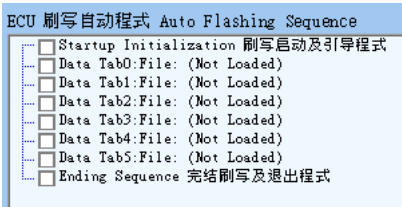
5.8.5. Managing .fcs file

The Bootloader commands in the View Tree will be stored in to a “fcs” file on exit of the Bootloader Panel, so that it can be re-initialised upon at restart of the program or can be reload it at any time as required per the designated ECU. The “fcs” stands for “Flash Command Sequence”. This fcs file is managed under the File Menu option as follows:



5.8.5.1. 新件 New (.fcs) ...

To Clear the commands in the Command tree and start a new project, below shows the new Command tree



5.8.5.2. 打开 Open (.fcs) ...

This is to open an existed fcs formatted file that has previously been saved.

5.8.5.3. 保存 Save (.fcs)

Save and override the current “Auto Flash Sequence” in to the currently opened fcs file

5.8.5.4. 另存为 SaveAs (.fcs) ...

Save the current “Auto Flash Sequence” in to the different fcs file

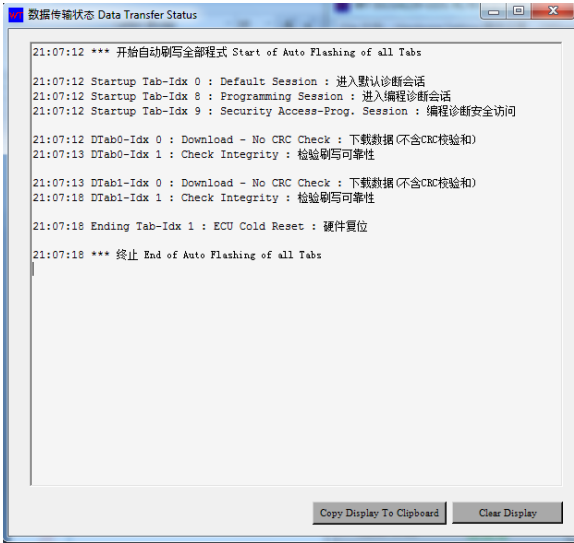
5.8.5.5. 退出 Exit

Exit and closed the “Bootloader Command Management” window.

5.8.6. Execute the “Auto Flash Sequence”

When all the command sequences are ready, click the Execute button **执行 Execute** complete the flashing of the ECU.

During the execution of the command sequence, a Window will be popped up showing the Step of the execution. Picture on the right hand side shows the sample.



Separate Window showing the status of the Auto Sequence

5.9. CAN device information

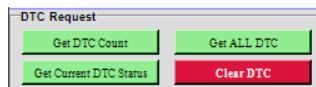
Hardware Type: 1 - Leaf : Leaf Light HS - 002418-13157 (Ch1)	Connection status : ON LINE	Baudrate: 500K	S/P: 62.5%	SJW: 1
--	-----------------------------	----------------	------------	--------

The above picture shows the CAN device information panel; which includes:

- The details of the selected Kvaser Hardware device
- CAN Bus connection status, i.e. On line or Off line
- CAN communication settings:
 - The bit rate (BAUD)
 - Sampling Point (S/P)
 - Synchronise Jump Width (SJW)

5.10. DTC status Commands and Controls

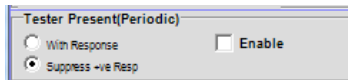
Picture below shows the 4 DTC command buttons available:



- Get DTC Count: Option 0x01; Reads the number of DTC
- Get Current DTC Status: Option 0x02; Reads current DTC Status
- Get ALL DTC: Option 0x0A – Reads all DTC regardless of the Status
- Clear DTC: Clear all or group of DTC; setting in the Ini file

5.11. “Tester Present” signal Controls

Picture below shows the control panels for “Tester Present” ; you can select the option for ECU to reply an Ack response or not, normally a response is not required. When ready, enable “Enable” check box to commence the periodic send out of the “Tester Present” command string.



Please note that the interval time to send out the “Tester Present” command string is set in the Ini file; the following example sets the interval time to 5 seconds: **%TesterPresentInterval=5000**

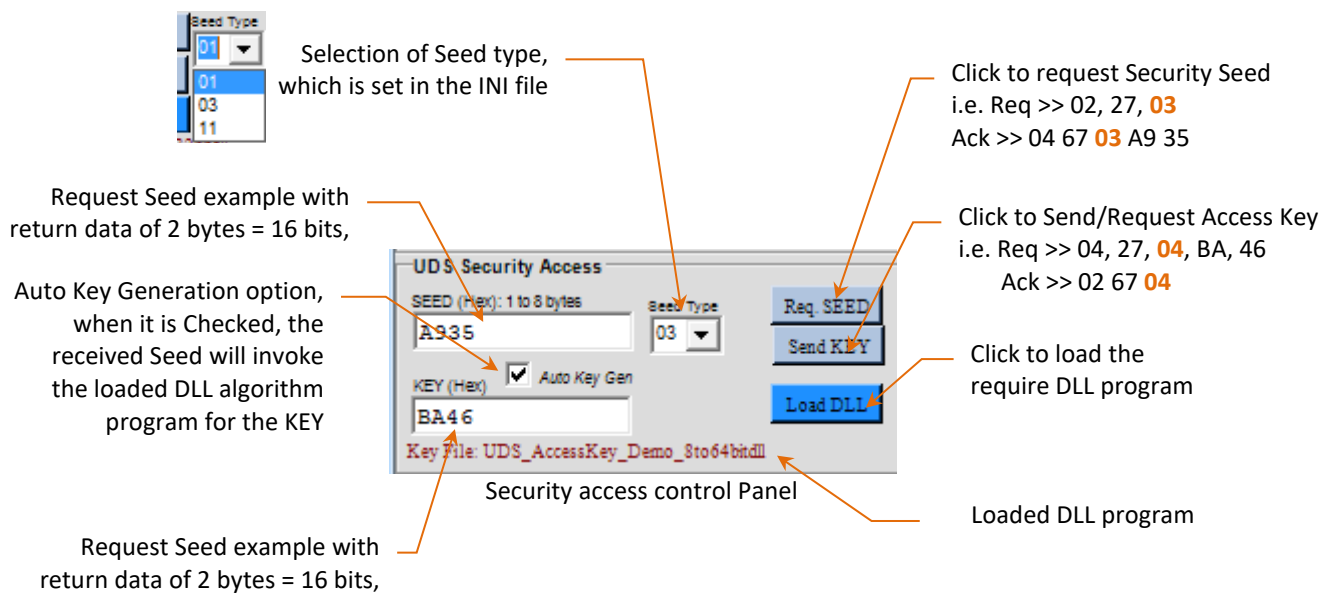
5.12. Security Access Controls

Security Key is generated via the UDS protocol with special algorithm to compute the given Seed to Key, the algorithm is confidential and varies from access to access, and therefore in order to protect the secrecy of the algorithm, we provide a sample DLL program for users to generate the embedded algorithm with ease and safe.

Following are available for key generation:

- Multiple Seed Type as per UDS recommendation
- Seed length can be 1 to 8 bytes long
- Generated Key can be 1 to 8 bytes long

Following diagram gives an over view of the control panel:



Example: Request Security Seed (32 bits) with Type 01

Req >> 02, 27, 01

Ack >> 06 67 01 F4 1F F4 1F

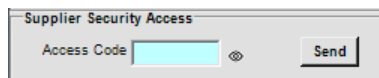
In the Send Key command, the type Seed value of F41FF41F will be Plus 01010101h and the Key will be F520F520h.

Example: Request Access Key (32 bits) for Type 01

Req >> 06, 27, 02, F5, 20, F5, 20

Ack >> 02 67 02

5.13. Password access- Supplier Specific



The picture above shows the panel of Supplier Security Access, this is Supplier Specific; the Command can be set in the Initialisation file.

5.14. ECU Reset buttons



The above picture shows the panel with two control buttons to reset ECU; they are as follows:

- <Hard Reset ECU>: Command 11h, 01h for battery (KL30) Off/On Reset
- <Key Off-On Reset>: Command 11h, 02h for ignition (KL15) Off/On reset

7 Appendix A: A quick start guide for preparation to Bootloader Operation

After the successful input of the Initialisation file, follows the following sequence to prepare for flashing the ECU:

Step 1: Click the **Online** button to enable the control panel for operation

Step 2: If the security DLL has not been loaded, click **Load DLL** to import the DLL file

Step 3: In the Tester Present (Periodic) panel, checked the ☐ **Enable** box to enable the period transmission of Tester Present

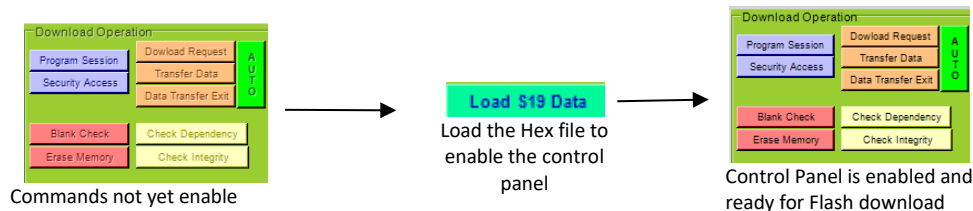
Step 4: Load the Hex data by clicking the **Load S19 Data** button; repeat this Hex file loading until all hex code are loaded.

Step 5: Now the Control panel is ready for downloading the new code to ECU for flashing/programming. See below Appendix B: Example of Re-flash Download for more details.

8 Appendix B: Example of Re-flash Download Operation

The embedded Bootloader function module in the ECU gets activated when the request for “Check Pre-Condition” as described in section 5.8.1.3 Application operation is returned with a positive response. In this instance, the ECU enters in to the Bootloader routines and ready for flashing New Program code. Please make sure the periodic “Tester Present” command is enabled to keep the ECU remains in the Bootloader routines.

When the Bootloader panel is loaded, the sub-panel “Download Operation” is disabled, but it will be enabled when the Hex Program file is loaded, following diagrams illustrate the sequence.



When the “Download Operation” panel is enabled, the flash download operation can commence.

- I. Start Programming section: Click this button **Program Session** for the ECU to start the Programming Session
- II. Enable Security Access: Click this button **Security Access** to activate the security access, please note that this demo has pre-set security algorithm, but user can enter specific Security Access algorithm by importing a DLL file containing the security algorithm.
- III. ECU ROM Memory Blank check: Click this button **Blank Check** to verify the ROM contents of the ECU are blank and ready for program download, the result of the request will be displayed on the interactive panel .
- IV. Erase ECU memory: Click this button **Erase Memory** to initiate the erasure process, when it is completed, a text message will be displayed on the display panel.
- V. Flash programming procedures: this operation requires 3 steps under the UDS recommendation:
 - ☐ Step 1: Request Download: Click this button **Download Request** to request the ECU to get ready for new program download. Upon successful request, a test message “Request for Download is successful!” will be shown at the bottom of the “Download Operation” panel.
 - ☐ Step 2: Transfer Data: Click this button **Transfer Data** to download the new program to ECU, the process of Memory Block downloading is shown on the bottom of the “Download Operation” panel e.g. **Tx=> 28800h:3E000h (64% Completed)** . Upon completion of successful download, a text message will be shown; including the time taken for the download.

Note: During the flash download operation, please do not use the mouse or activate other windows, this is because it will divert the PC resource for the new selected Window task and will affect the linear sequential timing of the Flash Download operation.

- ☐ Step 3: Request Transfer Exit: Click this button **Data Transfer Exit** to instruct the ECU to stop the Flash Download routine. The ECU will reply a positive response with a result (optional) of 8 bit sum checksums.

The AUTO button **AUTO** can be used to automate the 3 steps as described above.

- VI. Get Program CRC8/CRC16/CRC32: After the successful download operation, click this button **Check Dependency** to request the ECU to return a CRC8/CRC16/CRC32 checksums for the new downloaded program.
- VII. Validate the program integrity: In this example, the program has internal 16 bit Sum checksums for its own internal security check, click this button **Check Integrity** to request the validity of this 16 bit Sum Checksums. At this point, the Flash Download operation of a new program has been successfully carried out.

The ECU is ready to run the new program, you can simply click **Hard Reset ECU** button or apply Power On reset.

9 Appendix C: Sample of setting in INI File for Bootloader controls

```
%OBDSERVICEType=UDS
%Title=RL78 ECU CAN Demo-Generic
%TesterTxID=71B
%TesterTxIDAlt=7DF // Broadcast ID
%TesterRxID=79B
%DiagMode=01,DEFAULT
%DiagMode=02,PROGRAMMING
%DiagMode=03,EXTENDED
%DiagMode=61,SUPPLIER
%TesterPresentInterval=5000 // Tester Present Interval time 5 seconds
%SeperateTime=1000 // Separation time for the Multi-Frame
%Bootloader=UDS CAN- ECU Flash Programming
%SupplierEOL=Enable // Enable Supplier controls

//*****
//*** Startup Screen setting
//*****
%ShowBootloaderPanel=True // Show Bootloader Panel at startup

//*****
//*** General Setting
//*****
%BLDataSizeCountFrom1=False // Bootloader function- Memory size count from 1

//*****
//*** Security Access data
//*****
%RequestSeedType=01 // Provide Type 01-02 Option
%RequestSeedType=03 // Provide Type 03-04 Option
%RequestSeedType=05 // Provide Type 05-06 Option
%RequestSeedType=09 // Provide Type 09-0A Option
%RequestSeedType=11 // Provide Type 11-12 Option
%RequestSeedType=13 // Provide Type 13-14 Option
%RequestSeedType=15 // Provide Type 15-16 Option
%DLTestButton=Disable // "Enable" to enable the Test button to test the Seed-Key, others is disabled

//*****
//***DID Codes
//*****
%DIDEraserMemory=FF00 // DID for Erase Memory
%DIDMemoryBlankCheck=FE00 // DID for Memory Blank Check
%DIDCheckPreCondition=FF02 // DID for Pre-Condition inquiry
%DIDCheckProgramIntegrity=F001 // DID for Program Integrity Check
%DIDGetAppDependency=FF01 // DID for Application Dependency self check inquiry
%DIDGetBootloaderID=F180 // DID for Read the Bootloader version number/ID
%DIDUpdateFlashProgDateSupplier=F199 // DID for Updating the Supplier programming date detail
%DIDUpdateFlashProgDateOEM=F15A // DID for Updating the OEM programming date detail
%DIDUpdateFlashProgDateOEMAlt=F15B // DID for Updating an alternate OEM programming date detail

//*****
//***** CRC Masks
//*****
// CRC-Type, Polynomial(h), Initial Value(h), Final XOR Value(h), Reverse Data Byte(Boolean), Reverse CRC Result Before Final Xor(Boolean)
%CRC=8,07,00,00,False,False // CCITT-8
%CRC=16,1021,FFFF,0000,False,False // CRC-CCITT (16 bit)
%CRC=32,04C11DB7,FFFFFFFF,FFFFFFFF,True,True // CCITT-32
%CRCStartUp=CRC16 // Select CRC16 as Start-up CRC

//*****
//***** DTC Masks
//*****
%DTCReadStatusCountMask=FF
%DTCReadCurrentStatusMask=FF
%DTCClearInfoByStatusMask=FF,FF,FF

//*****
//***** Start Routine 3lh: return codes
//*****
%RoutineUseStartReplyAsResult=False // True = Use the response from StartRoutine as results;
// False= Use the normal 3 steps of StartRoutine procedures
%RoutineStartResultOK =0 // Byte value for OK response
%RoutineCompleted =2 // Byte value for Routine Complete
%RoutineInProgress =3 // Byte value for Routine still in progress
%RoutineStopped =4 // Byte value for Routine has stopped/Completed
%RoutineFailureOrNotRun =5 // Byte value for Routine failure or not run
%RoutineFnResultOK =0 // Byte value for Routine Result= OK
%RoutineFnResultFail=1 // Byte value for Routine Result= Fail

//*****
//***** Data Transfer
//*****
//%TransferExit8bitSumCSum=On // This is the 8 bit Non-Inverted Checksums for Data Block Transfer
%TransferExit8bitSumCSumInvert=On // This is the 8 bit Inverted Checksums for Data Block Transfer

//*****
//*****Command Strings
//*****
%CmdStrProgramModeAccessType=27,09 // UDS Command String for Security Access type (09) for Programming session Mode
%CmdStrExtendedModeAccessType=27,01 // UDS Command String for Security Access type (01) for Extended session Mode

%CmdStrGetBootloaderID=22,DID // UDS Command String for Read Bootloader version/ID
%CmdStrCheckPreCondition=31,01,DID // UDS Command String for Pre-Condition inquiry

%CmdStrEraseMemory=31,01,DID,StartAddr,Size // UDS Command String for Erase Memory
%CmdStrMemoryBlankCheck=22,DID,StartAddr,Size // UDS Command String for Memory Blank Check

%CmdStrDownloadRequest=34,00,44,StartAddr,Size // UDS Command String for Download Request
%CmdStrDownloadDataXfer=36 // UDS Command String for Data Transfer
%CmdStrDownloadXferExit=37 // UDS Command String for Data Transfer Exit
```

```
%CmdStrCheckProgramIntegrity=31,01,DID,TotalChksums // UDS Command String for Program Integrity Check
%CmdStrGetAppDependency=31,01,DID,StartAddr,Size,Chksums // UDS Command String for Application Dependency inquiry

%CmdStrUpdateFlashProgDateSupplier=2E,DID,YYYYMMDD // UDS Command String for System Supplier to program the Date Code
%CmdStrUpdateFlashProgDateOEM=2E,DID,YYMMDD,TesterID // UDS Command String for OEM to program the Date Code + Tester ID
%CmdStrReadFlashProgDateOEM=22,DID // UDS Command String for Read back the OEM Program date code details
```