How to integrate an automatic detection of MAP file names

Version 1.0.0
09/28/10

Application Note AN-IMC-1-010

Author(s) Ds
Restrictions Public Document - no restrictions - suitable for web and publication

Abstract This application note describes the integration of the automatic detection of MAP file names into the Vector XCP driver.

Revision history

Version	Author	Description
1.0	Ds	Documentation started

Table of Contents

1.0 Overview

With the help of a MAP file, CANape can automatically update the addresses of calibration and measurement objects in the A2L file. The advantage of retrieving its file name from the ECU is to avoid working with the wrong MAP file, or loading the files in the wrong order.

The aim of this document is to demonstrate the integration of the automatic detection of MAP file names into the Vector XCP driver. This is done by giving detailed information as well as a step by step instruction which shows how to integrate the functionality into the XCP driver of a simulated ECU.

An example with source code is part of the XCPsim project of the XCP_BASIC_DRIVER package. It is contained in the subdirectory ".\Samples\XCPsim" of the installation directory of this package.

Note: The screen shots in this application note are from CANape V9.0.

The Vector XCP driver is delivered with four files:

xcpProf.c	: Vector XCP driver source code
xcpProf.h	: Header file of the Vector XCP driver
xcp_cfg.h	: File to configure the Vector XCP driver
xcp_par.h	: File to configure the Vector XCP driver

The free XCP driver is delivered with the following three files:

xcpBasic.c	: Vector basic XCP driver source code
xcpBasic.h	: Header file of the Vector basic XCP driver
xcp_cfg.h	: File to configure the Vector basic XCP driver
xcp_par.h	: File to configure the Vector basic XCP driver

The Vector XCP driver is configured by editing the *xcp_cfg.h* file, which contains the definition for the automatic detection of MAP file names.

1.1 Configuring the XCP Driver

The *xcp_cfg.h* file is responsible for the configuration of the Vector XCP driver and must be adjusted manually for the specific ECU.

First of all, the option must be enabled. For the basic XCP driver implementation, this is done by editing the 'XCP_ENABLE_VECTOR_MAPNAMES' 'define' statement in the *xcp_cfg.h* file:

```
…
/*----------------------------------------------------------------------------*/
/* XCP parameters */
/* Enable the automatic detection of MAP names */


#define XCP_ENABLE_VECTOR_MAPNAMES /* e.g. xcpsim.map*/
```

1.2 Writing the Interface Function for the automatic detection

The function ‘ApplXcpGetIdData’ is implemented in the *xcpsim.c* file and is called by the xcp driver. It is responsible for building the character array containing the MAP format, the counter and the MAP file name. Additionally, it calculates the size of this array and the number of bytes which need to be uploaded by CANape.

In the following an excerpt from the delivered *xcpsim.c* file is shown:

```
// ---------------------------------------------------------------------------------
// ApplXcpGetIdData()
// Callback for xcpProf.c or xcpBasic.c
// ---------------------------------------------------------------------------------
#define MAP_FORMAT 29
#define MAP_NAME   "xcpsim"

unsigned char MapTest[500];
unsigned int MapTestSize;

vuint32 ApplXcpGetIdData( vuint8 **pData) {

   MapTestSize = sprintf((char*)MapTest,"%c%c%s.map",MAP_FORMAT,0,MAP_NAME);
  // Result: MapTest = ”290xcpsim.map\0”

  *pData = MapTest;

  return MapTestSize;
}
// --------------------------------------------------------------------------------
```

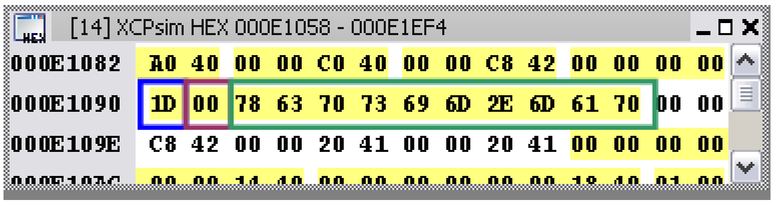
‘MAP_FORMAT’ represents the format of the MAP file. (See the table below).

‘0’ is the counter which is used by the master in the following communication sequence as address extension to define the corresponding memory address in the ECU. The range of the counter is from zero to 255. (See the ‘SHORT_UPLOAD’ example).

‘MAP_NAME’ is the name of the MAP file including the MAP extension and ‘\0’ which marks the end of the block.

For example:

‘290xcpsim.map\0’: MAP_FORMAT = 29 → ‘Microsoft standard’; Counter = 0; MAP_NAME = XCPSIM.MAP\0



```
[14] XCPsim HEX 000E1058 - 000E1EF4
000E1082  A0 40 00 00 C0 40 00 00 C8 42 00 00 00 00
000E1090  1D 00 78 63 70 73 69 6D 2E 6D 61 70 00 00
000E109E  C8 42 00 00 20 41 00 00 20 41 00 00 00 00
000E10AC  00 00 14 40 00 00 00 00 00 00 18 40 01 00
```

Table of MAP file formats:

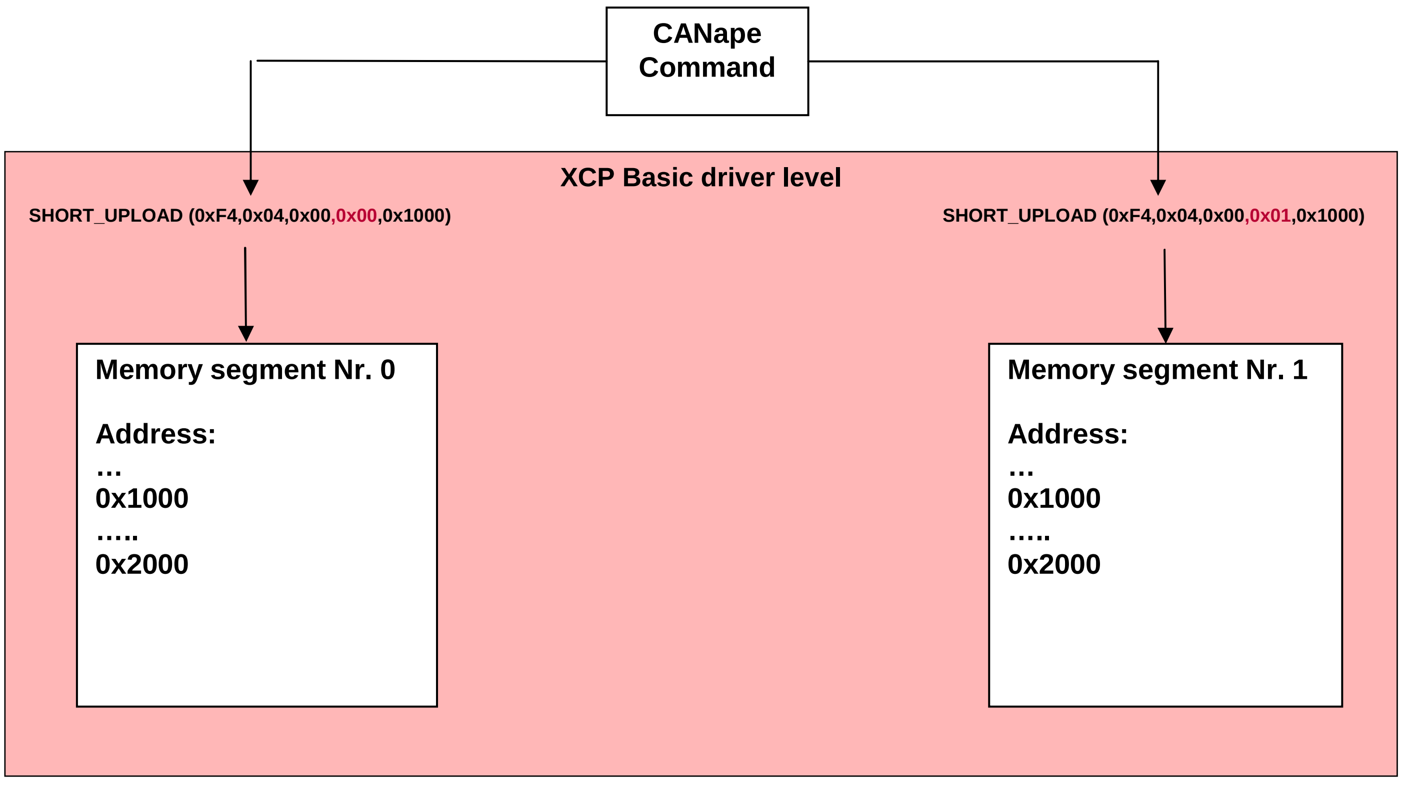
1 =	"BorlandC 16 Bit"
2 =	"M166"
3 =	"Watcom"
4 =	"HiTech HC05"
6 =	"IEEE"
7 =	"Cosmic"
8 =	"SDS"
9 =	"Fujitsu Softune 1(.mp1)"
10 =	"GNU"
11 =	"Keil 16x"
12 =	"BorlandC 32 Bit"
13 =	"Keil 16x (static)"
14 =	"Keil 8051"
15 =	"ISI"
16 =	"Hiware HC12"
17 =	"TI TMS470"
18 =	"Archimedes"
19 =	"COFF"
20 =	"IAR"
21 =	"VisualDSP"
22 =	"GNU 16x"
23 =	"GNU VxWorks"
24 =	"GNU 68k"
25 =	"DiabData"
26 =	"VisualDSP DOS"
27 =	"HEW SH7055"
28 =	"Metrowerks"
29 =	"Microsoft standard"
30 =	"ELF/DWARF 16 Bit"
31 =	"ELF/DWARF 32 Bit"
32 =	"Fujitsu Softune 3..8(.mps)"
33 =	"Microware Hawk"
34 =	"TI C6711"
35 =	"Hitachi H8S"
36 =	"IAR HC12"
37 =	"Greenhill Multi 2000"
38 =	"LN308(MITSUBISHI) for M16C/80"
39 =	"COFF settings auto detected"
40 =	"NEC CC78K/0 v35"
41 =	"Microsoft extended"
42 =	"ICCAVR"
43 =	"Omf96 (.m96)"
44 =	"COFF/DWARF"
45 =	"OMF96 Binary (Tasking C196)"
46 =	"OMF166 Binary (Keil C166)"
47 =	"Microware Hawk Plug&Play ASCII"
48 =	"UBROF Binary (IAR)"
49 =	"Renesas M32R/M32192 ASCII"
50 =	"OMF251 Binary (Keil C251)"
51 =	"Microsoft standard VC8"
52 =	"Microsoft VC8 Release Build (MATLAB DLL)"
53 =	"Microsoft VC8 Debug Build (MATLAB DLL)"
54 =	"Microsoft VC8 Debug file (pdb)"

Example for the usage of the MAP counter: SHORT_UPLOAD from slave to master

Category	Standard, optional
Mnemonic	SHORT_UPLOAD

Position	Type	Description
0	BYTE	Command Code = 0xF4
1	BYTE	Number of data elements [AG] [1..MAX_CTO-1]
2	BYTE	Reserved
3	BYTE	Address extension
4..7	DWORD	Address

A data block of the specified length, starting at the specified address will be returned. The MTA pointer is set to the first data byte behind the uploaded data block.

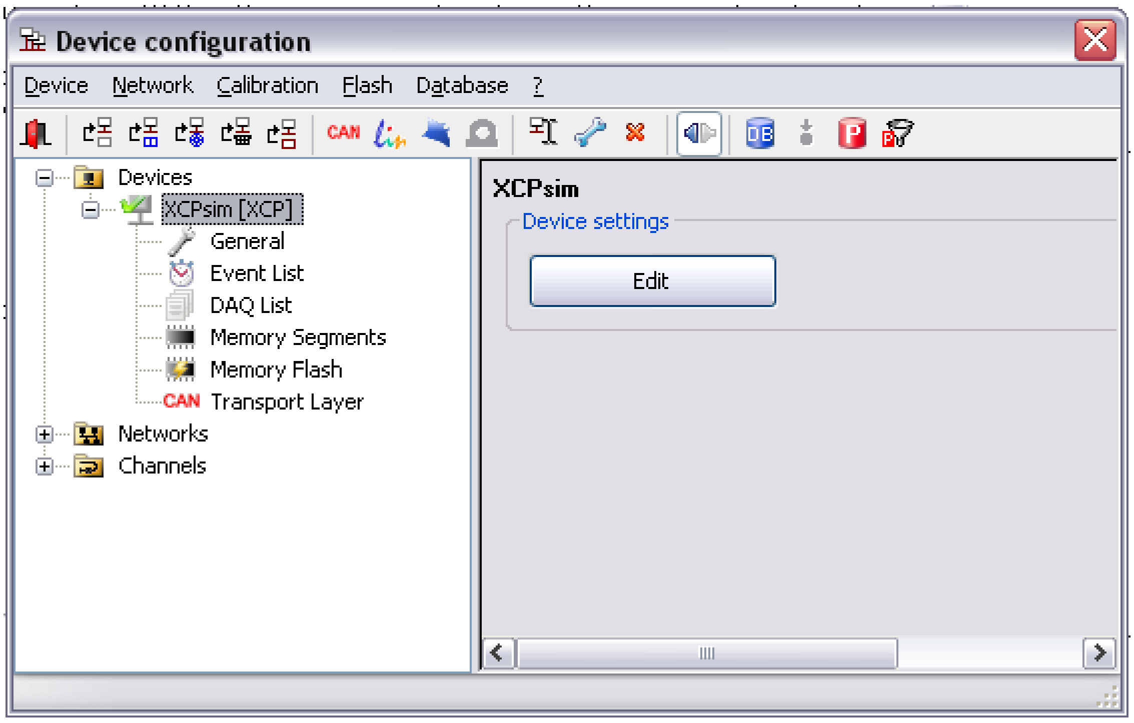


The command can have the same address and byte length but it must have a different address extension – the MAP counter - (red) to identify the appropriate memory block in the ECU.

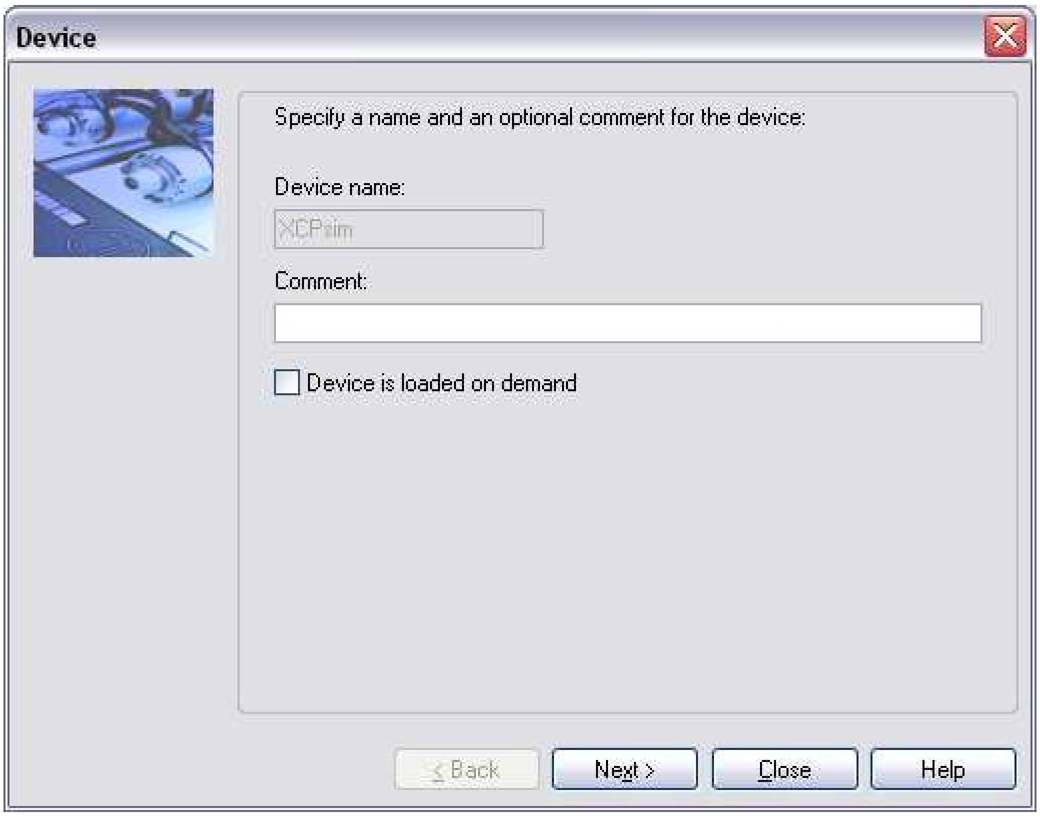
2.0 Enabling the automatic detection of the MAP file name in CANape

The automatic detection of the MAP file name needs to be activated in the ‘Device settings’ dialog of the ECU:

- Load the CANape project
- Select ‘Device’ | ‘Device configuration…’ in the main menu
- Select the device in the tree on the left-hand side and push the ‘Edit’ button



The settings dialog of the device opens up:



Specify a name and an optional comment for the device:

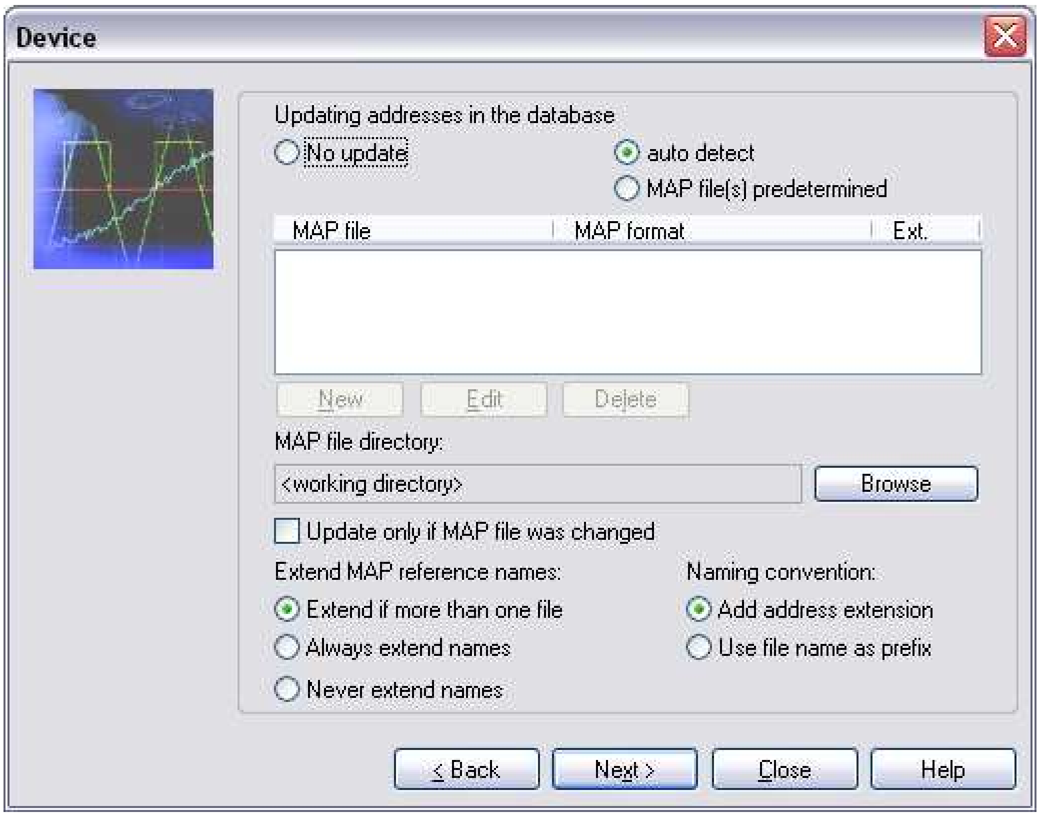
Device name:

XCPsim

Comment:

☐ Device is loaded on demand

After three clicks on the 'Next' button, the following dialog box appears:


Device

Updating addresses in the database

( ) No update (●) auto detect ( ) MAP file(s) predetermined

MAP file	MAP format	Ext.

MAP file directory: <working directory>

[ ] Update only if MAP file was changed

Extend MAP reference names: (●) Extend if more than one file ( ) Always extend names ( ) Never extend names

Naming convention: (●) Add address extension ( ) Use file name as prefix

Select 'auto detect'.

If this setting is active, CANape sends the following automatic detection sequence to the ECU when the connection is established, in order to retrieve the MAP file name from the ECU.

3.0 Automatic detection sequence

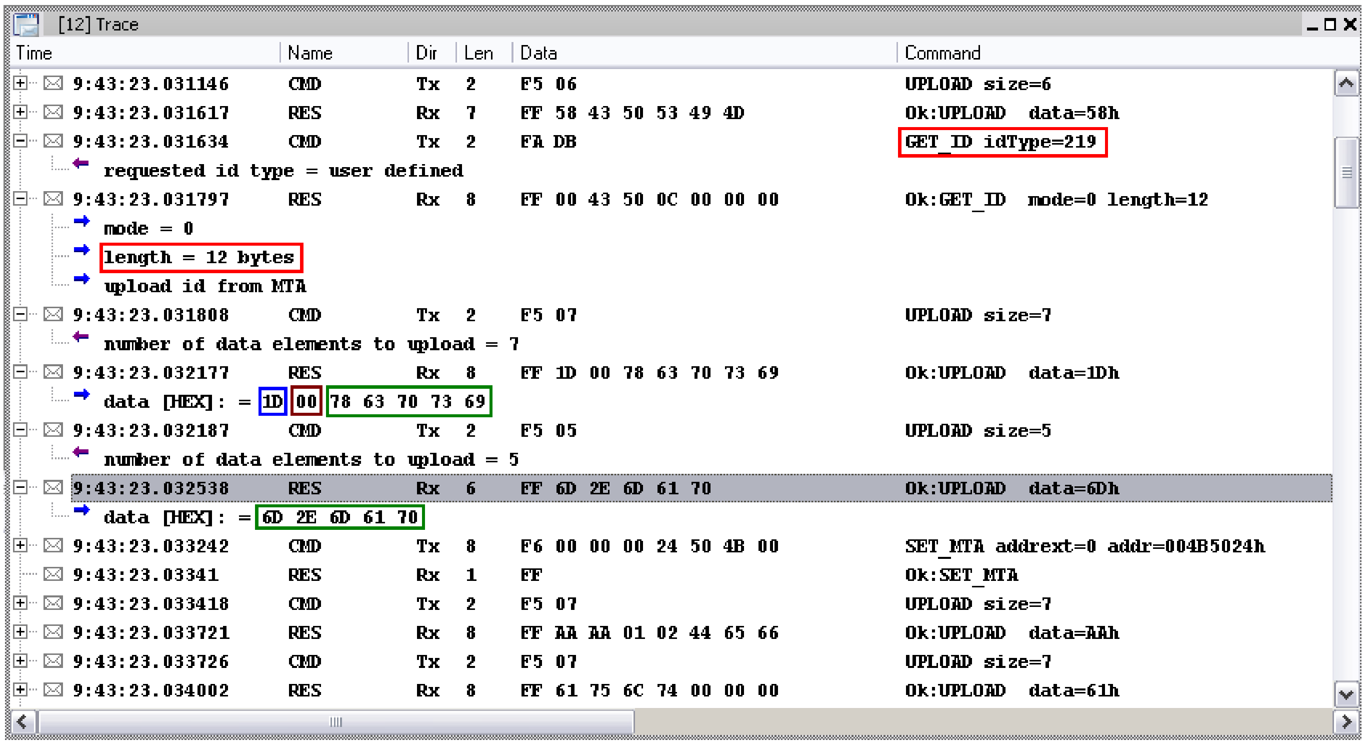
- The master (CANape) sends the command GET_ID with the idType = 219 (0xDB hex) to the ECU
- The ECU sets the MTA (memory transfer address) pointer to the first byte of the data block containing the MAP file identification and returns the number of bytes which need to be uploaded to the master as well as the counter which defines the appropriate memory segment
- The master requests the specified number of bytes from the ECU by sending UPLOAD commands
- The master looks for the appropriate MAP file in the MAP file directory and updates the addresses in the a2l file accordingly

Command sequence GET_ID

Id of the MAP format

Counter

MAP name



```
[12] Trace
Time               Name      Dir  Len  Data                        Command
9:43:23.031146     CMD       Tx   2    F5 06                       UPLOAD size=6
9:43:23.031617     RES       Rx   7    FF 58 43 50 53 49 4D        Ok:UPLOAD  data=58h
9:43:23.031634     CMD       Tx   2    FA DB                       GET_ID idType=219
    requested id type = user defined
9:43:23.031797     RES       Rx   8    FF 00 43 50 0C 00 00 00     Ok:GET_ID  mode=0 length=12
    mode = 0
    length = 12 bytes
    upload id from MTA
9:43:23.031808     CMD       Tx   2    F5 07                       UPLOAD size=7
    number of data elements to upload = 7
9:43:23.032177     RES       Rx   8    FF 1D 00 78 63 70 73 69     Ok:UPLOAD  data=1Dh
    data [HEX]: = 1D 00 78 63 70 73 69
9:43:23.032187     CMD       Tx   2    F5 05                       UPLOAD size=5
    number of data elements to upload = 5
9:43:23.032538     RES       Rx   6    FF 6D 2E 6D 61 70           Ok:UPLOAD  data=6Dh
    data [HEX]: = 6D 2E 6D 61 70
9:43:23.033242     CMD       Tx   8    F6 00 00 00 24 50 4B 00     SET_MTA addrext=0 addr=004B5024h
9:43:23.03341      RES       Rx   1    FF                          Ok:SET_MTA
9:43:23.033418     CMD       Tx   2    F5 07                       UPLOAD size=7
9:43:23.033721     RES       Rx   8    FF AA AA 01 02 44 65 66     Ok:UPLOAD  data=AAh
9:43:23.033726     CMD       Tx   2    F5 07                       UPLOAD size=7
9:43:23.034002     RES       Rx   8    FF 61 75 6C 74 00 00 00     Ok:UPLOAD  data=61h
```

4.0 Contacts

Vector Informatik GmbH Ingersheimer Straße 24 70499 Stuttgart Germany Tel.: +49 711-80670-0 Fax: +49 711-80670-111 Email: info@vector.com	**Vector CANtech, Inc.** 39500 Orchard Hill Pl., Ste 550 Novi, MI 48375 USA Tel: +1-248-449-9290 Fax: +1-248-449-9704 Email: info@vector-cantech.com	**VecScan AB** Lindholmspiren 5 402 78 Göteborg Sweden Tel: +46 (0)31 764 76 00 Fax: +46 (0)31 764 76 19 Email: info@vecscan.com
Vector France SAS 168 Boulevard Camélinat 92240 Malakoff France Tel: +33 (0)1 42 31 40 00 Fax: +33 (0)1 42 31 40 09 Email: information@vector-france.fr	**Vector Japan Co. Ltd.** Seafort Square Center Bld. 18F 2-3-12, Higashi-shinagawa, Shinagawa-ku J-140-0002 Tokyo Tel.: +81 3 5769 6970 Fax: +81 3 5769 6975 Email: info@vector-japan.co.jp	