CANape CCP Communication

Version 1.1
02/06/03

Application Note AN-AMC-1-100

Author(s)	Kim Lemon
Restrictions	Public Document

Abstract: This application note concentrates on explaining the fundamental concepts about CANape and CCP communication. Please check with Vector for any possible updates.

Table of Contents

1.0 Overview

The purpose if this document is to describe the CCP communication that occurs for various functions performed in CANape v3.1.

2.0 Application Note Content

This document focuses on establishing a CCP connection, calibration sequences, and flash programming communications. A windshield wiper simulator program for a Motorola HC12GD128 and the CCPSim programs were used to create the CAN messages described in this document.

Abbr.	
CAN	Controller Area Network
CCP	CAN Calibration Protocol
ASAM	Working Group of the Standardization of …
ECU	Electronic Control Unit

2.1 Summary Of CCP Commands Used With CANape

The following table lists the CCP v2.1 commands, and a brief description of when they are sent by CANape.

Command	Code	Usage
CONNECT	0x01	Initial command sent to establish a connection with the slave device (ECU). CANape sends this command to establish the online state with the ECU.
GET_CCP_VERSION	0x1B	CANape sends this command after the CONNECT command to verify that CANape is using the same CCP version as the ECU for communication. This command is sent prior to the EXCANGE_ID.
GET_SEED	0x12	CANape is sent by when the Seed and Key option is enable in CANape and CANape needs to determine if calirbartion, data acquistion, or flash programming is protected. CANape can then send the UNLOCK command. Only one resource or function may be requested with one GET_SEED command, the command returns 'seed' data for a seed & key algorithm. This is an optional CCP command
UNLOCK	0x13	CANape sends this command to unlock the slave's security protection using a 'key' computed from 'seed'. This is an optional CCP command.
BUILD_CHKSUM	0x0E	CANape uses the checksum on startup to determine if the ECU calibration values matches the values last saved by CANape. If the checksum matches, CANape will not have to upload the calibration data from the ECU. This is an optional CCP command.
EXCHANGE_ID	0x17	The EXCHANGE_ID is used by CANape to match the controller description file to the name defined in the ECU. The UPLOAD command is to get the device ID defined in the ECU.
UPLOAD	0x04	CANape uses this command to upload data from the ECU. For example, this is used to upload the Device ID and for uploading flash data from the ECU when requested by the user.
SET_ MTA	0x02	This sets the memory transfer address (MTA) to the start address of the data. This is usually used by CANape prior to an upload or download command.
SELECT_CAL_PAGE	0x11	If the Page Switching feature has been activated in CANape, then CANape will send the active cal page (Flash or RAM) prior to a SET_MTA and the UPLOAD or DOWNLOAD. This is an optional CCP command.
DNLOAD	0x03	Used by CANape to send RAM data to the ECU. This command allows up to five data bytes to be transferred. One example would be sending new characteristic map values to the ECU.
GET_DAQ_SIZE	0x14	Used by CANape when user has requested the start of measurement to determine the number and size of the DAQ lists in the ECU.
SET_DAQ_PTR	0x15	Used by CANape when user has requested the start of measurement to determine setup the ODT elements within a DAQ.
WRITE_DAQ	0x16	Used by CANape when user has requested the start of measurement to determine define the address assign to the ODT elements within a DAQ. A SET_DAQ_PTR is sent prior to this command.
START_STOP	0x06	Used by CANape when user has requested the start/stop of a measurement. The three modes for this command are Stop, Start, and prepare DAQ for synchronize start. With multiple DAQ events defined, CANape will use the START_STOP_ALL for the start and stop command, and only prepare DAQ mode will be used for this

		command.
START_STOP_ALL	0x08	Used by CANape when user has requested the start/stop of a measurement. This is an optional CCP command.
SHORT_UP	0x0F	This command is used to upload data from the ECU. The address is defined in the command, so no SET_MTA is required prior to this command. CANape uses this to upload calibration data and for polled measurement data. Up to five bytes of data can be retrieved with one command. This is an optional CCP command
CLEAR_MEMORY	0x10	CANape sends this command at the start of the flash programming sequence, so the ECU can clear the Flash sector. This is an optional CCP command.
PROGRAM_6	0x22	CANape sends this command during a flash programming sequence, so the ECU can program six bytes of data. This is an optional CCP command.
PROGRAM	0x18	CANape sends this command during a flash programming sequence, so the ECU can program one to five bytes of data. This is an optional CCP command.
DNLOAD_6	0x23	Used by CANape to send RAM data to the ECU. This command allows six bytes to be transferred. One example would be sending a flash kernel to the ECU. This is an optional CCP command.
GET_ACTIVE_CAL_PAGE	0x09	This command returns the start address of the calibration page currently active in the slave device. This is an optional CCP command.
TEST	0x05	The command is used to determine if the ECU is available for CCP communication. This command is sent by CANape using the “Test Connection” button in the device configuration window. This is an optional CCP command.
DISCONNECT	0x07	This command is used to end a session with a particular slave temporarily or to end the communication indefinitely. CANape sends an end of session when switched from the online to offline state.
SET_S_STATUS	0x0C	This command keeps the slave informed about the current state of the calibration session. This is an optional CCP command.
GET_S_STATUS	0x0D	This is an optional CCP command.
MOVE	0x19	This is an optional CCP command.
DIAG_SERVICE	0x20	This is an optional CCP command.
ACTION_SERVICE	0x21	This is an optional CCP command.

2.2 Connection Command Sequence

The following commands show the commands sequence that can occur when CANape establishes a connection with an ECU. The connection sequence will vary based on the CCP options implemented in the ECU and set in CANape.

Connected Command:

```
667            Rx   d 8 01 01 39 00 00 00 00 00

7E1            Rx   d 8 FF 00 01 FE 39 00 00 00
```

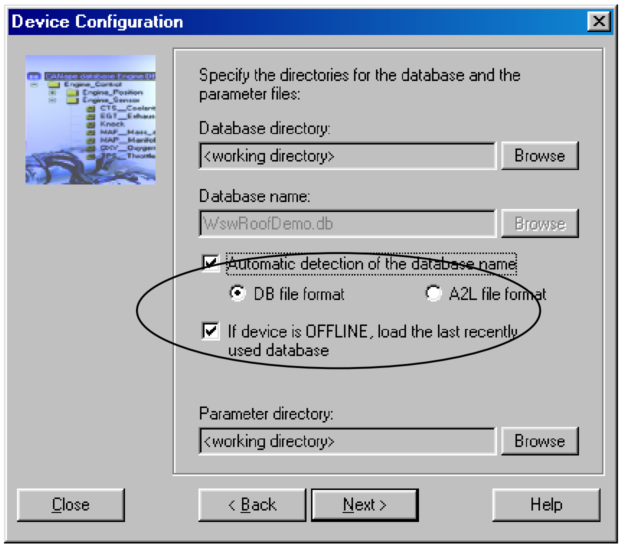
CCP Get Version Command:

```
667            Rx   d 8 1B 02 02 01 00 00 00 00

7E1            Rx   d 8 FF 00 02 02 01 00 00 00
```

Exchange Id Sequence (Optional)

```
        Rx    d 8 17 03 00 01 00 00 00 00

        Rx    d 8 FF 00 03 0B 00 43 00 83

        Rx    d 8 04 04 05 01 00 00 00 00

        Rx    d 8 FF 00 04 57 73 77 52 6F

        Rx    d 8 04 05 05 01 00 00 00 00

        Rx    d 8 FF 00 05 6F 66 44 65 6D

        Rx    d 8 04 06 01 01 00 00 00 00

        Rx    d 8 FF 00 06 6F 66 44 65 6D
```

The Exchange Id Sequence is executed based on the Automatic Detection Setting in the Device Configuration. The Exchange Id command is followed by a series of Upload Commands to obtain the device name from the ECU.


Device Configuration

Specify the directories for the database and the parameter files:

Database directory: <working directory>

Database name: WswRoofDemo.db

[✓] Automatic detection of the database name

(●) DB file format ( ) A2L file format

[✓] If device is OFFLINE, load the last recently used database

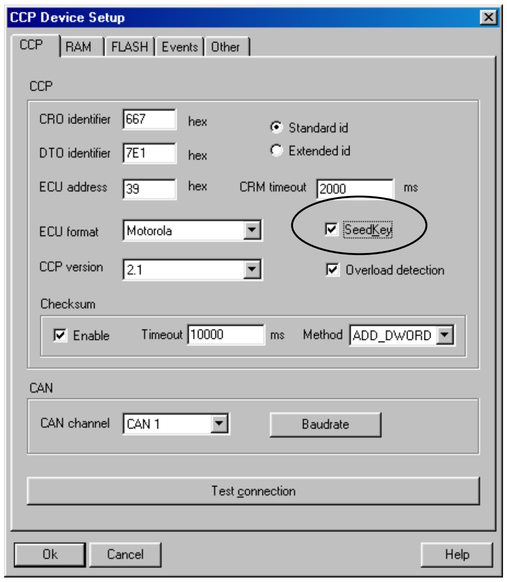
Parameter directory: <working directory>

Figure 1: Automatic Database Name Detection

Get Seed and Unlock Command Sequence (Optional)

```
        667              Rx    d 8 12 04 01 00 00 00 00 00

        7E1              Rx    d 8 FF 00 04 01 5A 01 00 00

        667              Rx    d 8 13 05 27 19 03 9D 00 00

        7E1              Rx    d 8 FF 00 05 01 5A 01 00 00

        667              Rx    d 8 12 06 02 00 00 00 00 00

        7E1              Rx    d 8 FF 00 06 01 9E 68 D3 00

        667              Rx    d 8 13 07 BD 27 2D 0F 00 00

        7E1              Rx    d 8 FF 00 07 03 9E 68 D3 00
```

The Seed and Key sequence is executed if the SeedKey option is selected in the Device Setup window.



CCP

CRO identifier 667 hex
DTO identifier 7E1 hex
ECU address 39 hex
[✓ Standard id] Extended id
CRM timeout 2000 ms
ECU format Motorola
[✓ SeedKey]
CCP version 2.1
[✓ Overload detection]

Checksum

[✓ Enable] Timeout 10000 ms Method ADD_DWORD

CAN

CAN channel CAN 1

Figure 2: Seed and Key Setting

Build Checksum (Optional)

```
667            Rx   d 8 02 0A 00 00 00 00 00 00
7E1            Rx   d 8 FF 00 0A 4D 43 50 53 49
667            Rx   d 8 11 0B 00 00 00 00 00 00
7E1            Rx   d 8 FF 00 0B 4D 43 50 53 49
667            Rx   d 8 02 0C 00 00 BC 0E 04 00
7E1            Rx   d 8 FF 00 0C 4D 43 50 53 49
667            Rx   d 8 0E 0D CD 06 00 00 00 00
7E1            Rx   d 8 FF 00 0D 02 5B 3F 00 00
```

The Build Checksum option is used to allow CANape to build a checksum value on the last saved calibration values. The CANape checksum matches the ECU checksum, and then CANape will populate the calibration windows with information from the last saved values. This allows CANape to not have to send Short_Up commands to retrieve the latest calibration data.

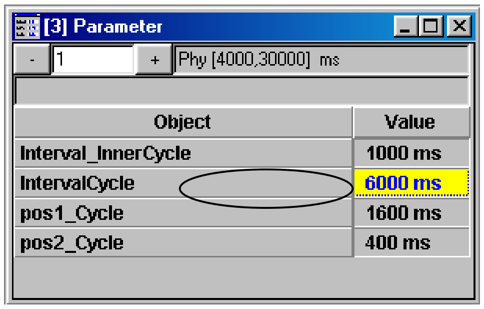
Short Upload Sequence

```
667            Rx   d 8 0F 07 02 00 00 00 25 00
7E1            Rx   d 8 FF 00 07 00 00 44 65 6D
667            Rx   d 8 0F 08 02 00 00 00 25 02
7E1            Rx   d 8 FF 00 08 00 00 44 65 6D
```

The short upload command is used by CANape to retrieve the latest parameter and map data from the ECU. The number of Short_Up commands sent would vary based on the number of parameters and maps displayed in the CANape configuration. The number of Short Up commands can also change if the checksum option is used.

2.3 Changing a Parameter Value

In this example, the Interval Cycle was originally 6000 and a value of 6000 was entered. The variable has a linear scale of 10 defined in the database. Therefore, the raw value sent to the ECU is 600 or 0x258. Page switching was not implemented and the database contains the RAM address of 0x2581 for the variable.



Object	Value
Interval_InnerCycle	1000 ms
IntervalCycle	6000 ms
pos1_Cycle	1600 ms
pos2_Cycle	400 ms

Figure 3: Parameter Data Changed

Set MTA:

```
        667            Rx   d 8 02 66 00 00 00 00 25 81
        7E1            Rx   d 8 FF 00 66 01 F4 44 65 6D
```

Download:

```
        667            Rx   d 8 03 67 02 02 58 00 00 81
        7E1            Rx   d 8 FF 00 67 01 F4 44 65 6D
```

Short Up

```
        667            Rx   d 8 0F 68 02 00 00 00 25 81
        7E1            Rx   d 8 FF 00 68 02 58 44 65 6D
```

The above sequence of commands is performed for all changes to the calibration data. The address for the parameter is sent to the ECU, the data for that address is sent to the ECU, and then CANape reads the data back from the ECU to verify the change.

2.4 Loading Parameter Data from a Saved File

A parameter window was selected in CANape. The Load command was selected from the popup menu. In this example, the Interval Cycle was originally 6000 and a value of 5000 was loaded from a saved file. The variable has a linear scale of 10 defined in the database. Therefore, the raw value sent to the ECU is 500 or 0x1F4. Page switching was not implemented and the database contains the RAM address of 0x2581 for the variable.

Set MTA:

```
        667            Rx   d 8 02 CB 00 00 00 00 25 81
        7E1              Rx   d 8 FF 00 CB 02 58 44 65 6D
```

Download:

```
        667              Rx   d 8 03 CC 02 01 F4 00 00 81
        7E1              Rx   d 8 FF 00 CC 02 58 44 65 6D
```

Short Up

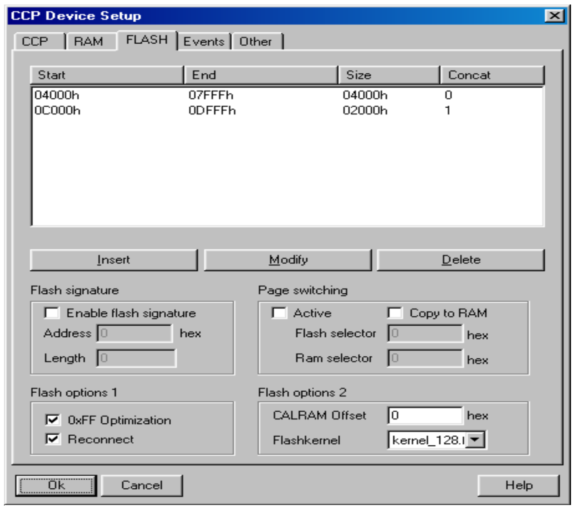
```
        667              Rx   d 8 0F CD 02 00 00 00 25 81
        7E1              Rx   d 8 FF 00 CD 01 F4 44 65 6D
        667              Rx   d 8 0F CE 02 00 00 00 25 7F
        7E1              Rx   d 8 FF 00 CE 00 64 44 65 6D
        667              Rx   d 8 0F CF 02 00 00 00 25 81
        7E1              Rx   d 8 FF 00 CF 01 F4 44 65 6D
        667              Rx   d 8 0F D0 01 00 00 00 25 8A
        7E1              Rx   d 8 FF 00 D0 00 F4 44 65 6D
```

CANape continues to send short upload commands to refresh all of the calibration values on the screen after loading parameter values from a file.

2.5 Upload File to Flash Program Sequence

From the CANape menu the Calibration | Upload file from Flash… was selected.

The following figure shows the flash programming settings in CANape for the Upload sequence. This example has multiple flash sectors.



Start	End	Size	Concat
04000h	07FFFh	04000h	0
0C000h	0DFFFh	02000h	1

Flash signature
- [ ] Enable flash signature
- Address: 0 hex
- Length: 0

Page switching
- [ ] Active
- [ ] Copy to RAM
- Flash selector: 0 hex
- Ram selector: 0 hex

Flash options 1
- [✓] 0xFF Optimization
- [✓] Reconnect

Flash options 2
- CALRAM Offset: 0 hex
- Flashkernel: kernel_128.l

Figure 4: Multiple Flash Sectors Setup

Set MTA:

```
        667             Rx    d 8 02 6C 00 00 00 00 40 00
        7E1             Rx    d 8 FF 00 6C 02 05 44 65 6D
```

Upload:

```
        667             Rx    d 8 04 6D 05 00 00 00 40 00
        7E1             Rx    d 8 FF 00 6D 00 01 02 03 04
        667             Rx    d 8 04 6E 05 00 00 00 40 00
        7E1             Rx    d 8 FF 00 6E 05 06 07 08 09
        667             Rx    d 8 04 6F 05 00 00 00 40 00
        7E1             Rx    d 8 FF 00 6F 0A 0B 0C 0D 0E
```

Upload commands are continued until the full flash sector has been uploaded.

According to the CCP v2.1 Specification an Upload command requires that the MTA0 pointer be post-incremented by the value of 'size' in the Upload command. Only one Set MTA is required at the start of the sequence. In the example, two flash sectors are defined. Therefore, after the first flash sector has completed upload, another Set MTA is issued to retrieve the next sector.

Set MTA:

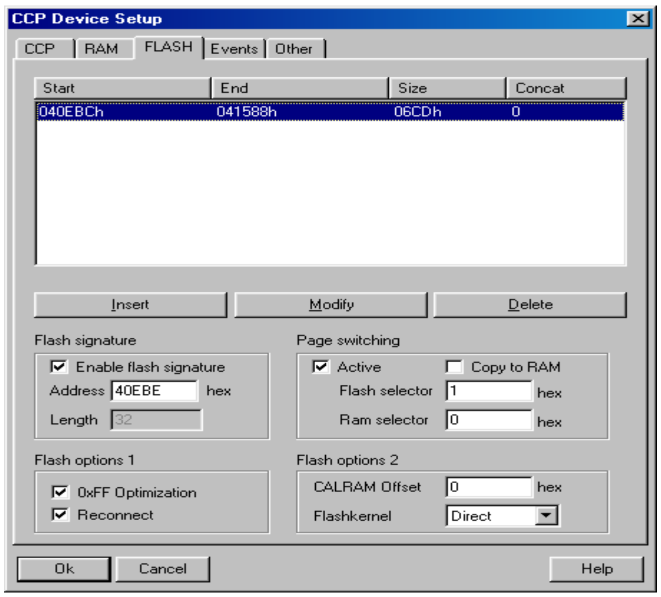
```
        667             Rx    d 8 02 3A 00 00 00 00 C0 00
        7E1             Rx    d 8 FF 00 3A FF FF FF FF FF
```

Upload:

```
        667             Rx    d 8 04 3B 05 00 00 00 C0 00
        7E1             Rx    d 8 FF 00 3B FF FF FF FF FF
        667             Rx    d 8 04 3C 05 00 00 00 C0 00
        7E1             Rx    d 8 FF 00 3C FF FF FF FF FF
```

Upload commands are continued until the full flash sector has been uploaded.

The following is another example with the page-switching feature enabled. The figure shows the flash programming settings in CANape for the Upload sequence.



Start	End	Size	Concat
040EBCh	041588h	06CDh	0

Flash signature
☑ Enable flash signature
Address 40EBE hex
Length 32

Page switching
☑ Active ☐ Copy to RAM
Flash selector 1 hex
Ram selector 0 hex

Flash options 1
☑ 0xFF Optimization
☑ Reconnect

Flash options 2
CALRAM Offset 0 hex
Flashkernel Direct

Figure 5: Single Flash Sector Setup

Set MTA:
```
        667          Rx   d 8 02 0E 00 00 01 00 00 00
        7E1          Rx   d 8 FF 00 0E 02 5B 3F 00 00
```

Select Cal Page:
```
        667          Rx   d 8 11 0F 00 00 01 00 00 00
        7E1          Rx   d 8 FF 00 0F 02 5B 3F 00 00
```

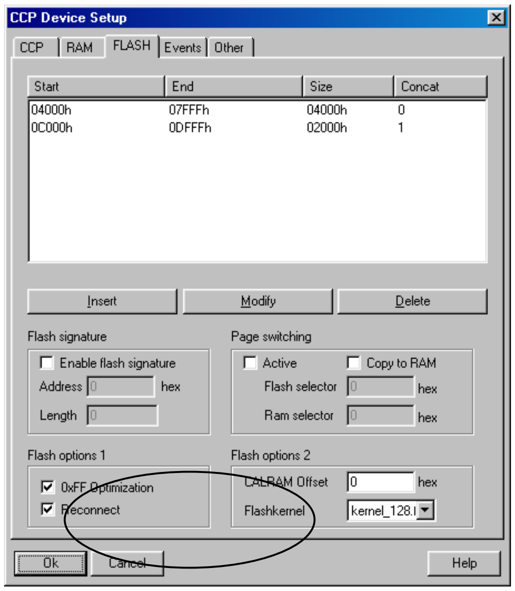
Set MTA:
```
        667          Rx   d 8 02 10 00 00 BC 0E 04 00
        7E1          Rx   d 8 FF 00 10 02 5B 3F 00 00
```

Upload:
```
        667          Rx   d 8 04 11 05 00 BC 0E 04 00
        7E1          Rx   d 8 FF 00 11 AA AA 74 65 73
        667          Rx   d 8 04 12 05 00 BC 0E 04 00
        7E1          Rx   d 8 FF 00 12 74 66 75 6C 6C
```

Upload commands are continued until the full flash sector has been uploaded.
The above example shows when the page switching is active, CANape sends a Set MTA command with the Flash Sector value specified on the Device Setup Flash page in the address section of the MTA command. In the above Example, a 1 was in the address portion of the Set MTA command. This command is followed by the Select Cal Page command. This sequence gives indication to the ECU that flash sector values are being requested from CANape.

2.6 Download File to Flash Program Sequence

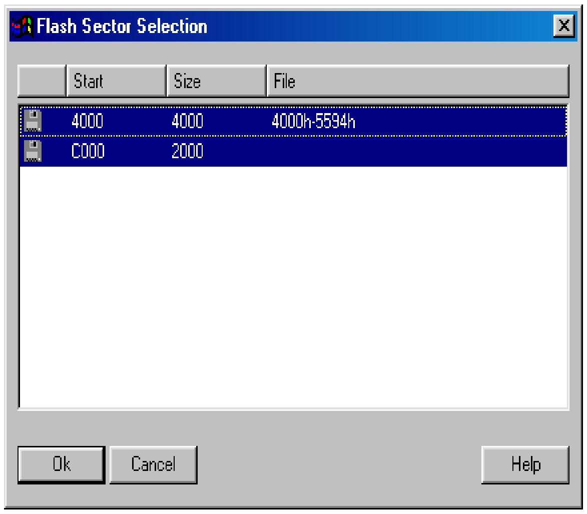
From the CANape menu the Calibration Download file to Flash was selected (**Calibration** → **Download file to Flash**). After the file is selected, CANape displays a Flash Sector Selection window based on the settings on the Device Setup Flash page. The following figures show the settings for this example.



Start	End	Size	Concat
04000h	07FFFh	04000h	0
0C000h	0DFFFh	02000h	1

Flash signature: ☐ Enable flash signature, Address 0 hex, Length 0
Page switching: ☐ Active, ☐ Copy to RAM, Flash selector 0 hex, Ram selector 0 hex
Flash options 1: ☑ 0xFF Optimization, ☑ Reconnect
Flash options 2: CALRAM Offset 0 hex, Flashkernel kernel_128.

Figure 6: Flash Settings



Start	Size	File
4000	4000	4000h-5594h
C000	2000	

Figure 7: Flash Sector Selection

This gives the user an opportunity to flash program all list sectors or specific sectors and allows for flashing of only the calibration sector (if the parameters in the ECU are setup to allow this).

Set MTA:
```
        667            Rx   d 8 02 63 00 00 00 00 20 00
        7E1            Rx   d 8 FF 00 63 02 05 44 65 6D
```

Program Prepare (Vector Extension to CCP to prepare for flash kernel download):
```
        667            Rx   d 8 1E 64 04 02 00 00 20 00
        7E1            Rx   d 8 FF 00 64 02 05 44 65 6D
```

Set MTA:
```
        667            Rx   d 8 02 65 00 00 00 00 20 00
        7E1            Rx   d 8 FF 00 65 02 05 44 65 6D
```

Download 6:
```
        667            Rx   d 8 23 66 3B 14 10 EC 84 7C
        7E1            Rx   d 8 FF 00 66 02 05 44 65 6D
        667            Rx   d 8 23 67 23 FE EC 80 7C 23
        7E1            Rx   d 8 FF 00 67 02 05 44 65 6D
```

The above sequences of commands are necessary because a flash kernel is being used. This can be seen in the Flash Settings figure. With this option, CANape downloads the flash routines to the RAM on the ECU. The location of the RAM is defined in the flash kernel file. The download sequence continues until the flash kernel is downloaded. There is a Download command (0x03) that can be used at the end of the sequence if less than six bytes need to be downloaded.

```
        667            Rx   d 8 02 11 00 00 00 00 21 3A
        7E1            Rx   d 8 FF 00 11 02 05 44 65 6D
        667            Rx   d 8 1F 12 00 00 06 67 07 E1
        7E1            Rx   d 8 FF 00 12 02 05 44 65 6D
        667            Rx   d 8 02 13 00 00 00 00 40 00
        7E1            Rx   d 8 FF 00 13 02 05 44 65 6D
        667            Rx   d 8 10 14 00 00 40 00 40 00
        7E1            Rx   d 8 FF 00 14 02 05 44 65 6D
        667            Rx   d 8 02 15 00 00 00 00 40 00
        7E1            Rx   d 8 FF 00 15 02 05 44 65 6D
        667            Rx   d 8 22 16 00 01 02 03 04 05
        7E1            Rx   d 8 FF 00 16 02 05 44 65 6D


        667            Rx   d 8 22 C0 00 00 06 00 00 FF
        7E1            Rx   d 8 FF 00 C0 02 05 44 65 6D
        667            Rx   d 8 18 C1 02 41 16 00 00 FF
        7E1            Rx   d 8 FF 00 C1 02 05 44 65 6D
        667            Rx   d 8 18 C2 00 41 16 00 00 FF
        7E1            Rx   d 8 FF 00 C2 02 05 44 65 6D
        667            Rx   d 8 01 C3 39 00 00 00 00 00
        7E1            Rx   d 8 FF 00 C3 FE 39 00 00 00
        667            Rx   d 8 17 C4 00 00 00 00 00 00
        7E1            Rx   d 8 FF 00 C4 0B 00 43 00 83
        667            Rx   d 8 04 C5 05 00 00 00 00 00
        7E1            Rx   d 8 FF 00 C5 57 73 77 52 6F
        667            Rx   d 8 04 C6 05 00 00 00 00 00
        7E1            Rx   d 8 FF 00 C6 6F 66 44 65 6D
        667            Rx   d 8 04 C7 01 00 00 00 00 00
        7E1            Rx   d 8 FF 00 C7 6F 66 44 65 6D
        667            Rx   d 8 0F C8 02 00 00 00 25 00
        7E1              Rx   d 8 FF 00 C8 00 00 44 65 6D
        667              Rx   d 8 0F C9 02 00 00 00 25 02
```

2.7 Loading Calibration Map Data from a Saved File

A calibration map was selected in CANape. The Load command was selected from the popup menu.

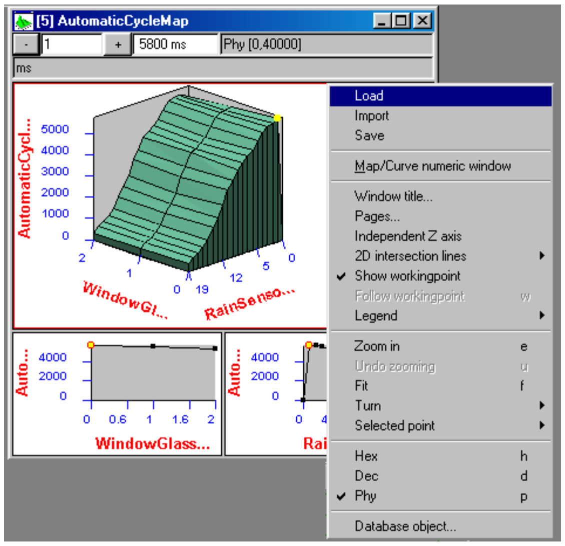


Figure 8: Load Calibration Map

Set MTA:

```
        667              Rx   d 8 02 5A 00 00 00 00 25 38
        7E1              Rx   d 8 FF 00 5A 41 16 00 00 FF
```

Download:

```
        667              Rx   d 8 03 5B 02 01 DC 00 00 38
        7E1              Rx   d 8 FF 00 5B 41 16 00 00 FF
```

Short Up:

```
        667              Rx   d 8 0F 5C 02 00 00 00 25 38
        7E1              Rx   d 8 FF 00 5C 01 DC 00 00 FF
```

Set MTA:

```
        667              Rx   d 8 02 5D 00 00 00 00 40 00
        7E1              Rx   d 8 FF 00 5D 01 DC 00 00 FF
```

Download:

```
        667              Rx   d 8 03 5E 01 00 00 00 00 00
        7E1              Rx   d 8 FF 00 5E 01 DC 00 00 FF
```

Short Up:

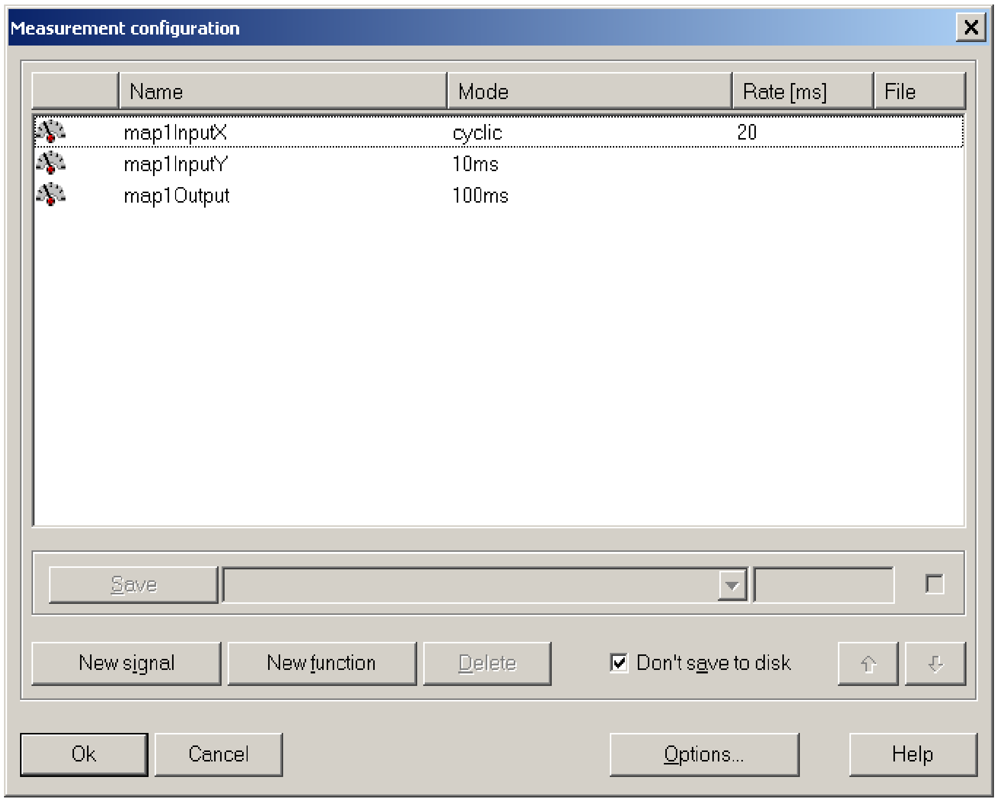
```
        667              Rx   d 8 0F 5F 01 00 00 00 40 00
        7E1              Rx   d 8 FF 00 5F 00 DC 00 00 FF
```

The sequence of setting the address, downloading the data, and uploading the same data from the ECU is repeated until the complete map is programmed. The Set MTA addresses are defined for the map in the database.

```
667             Rx   d 8 0F 56 02 00 00 00 25 00
7E1             Rx   d 8 FF 00 56 00 00 00 00 FF
667             Rx   d 8 0F 57 02 00 00 00 25 02
7E1             Rx   d 8 FF 00 57 00 00 00 00 FF
```

2.8 Measurement Start Sequence with DAQ Events

The CCPSIM program was configured with a short list of measurement data. The measurement data list contains three each being received at a different time internal. Therefore, three DAQ lists will be required to send the data.



	Name	Mode	Rate [ms]	File
	map1InputX	cyclic	20	
	map1InputY	10ms		
	map1Output	100ms		

Figure 9 - Measurement Setup

This is the sequence of CCP messages that occur have starting the measurement in CANape.
Determine size and number of DAQs.

```
667           Rx   d 8 14 6C 00 00 E1 07 00 00
7E1           Rx   d 8 FF 00 6C 0A 00 D1 FF FF
667           Rx   d 8 14 6D 01 00 E1 07 00 00
7E1           Rx   d 8 FF 00 6D 0A 0A D1 FF FF
667           Rx   d 8 14 6E 02 00 E1 07 00 00
7E1           Rx   d 8 FF 00 6E 0A 14 D1 FF FF
667           Rx   d 8 14 6F 03 00 E1 07 00 00
7E1           Rx   d 8 FF 00 6F 0A 1E D1 FF FF
667           Rx   d 8 14 70 04 00 E1 07 00 00
7E1           Rx   d 8 FF 00 70 0A 28 D1 FF FF
667           Rx   d 8 14 71 05 00 E1 07 00 00
7E1           Rx   d 8 FF 00 71 00 32 D1 FF FF
```

The GET_DATQ_SIZE command is repeated until the size of the ECU sends back a size of zero. CANape then knows the number of DAQs and the size of each in the ECU. The DAQ lists can then be setup by CANape. In the above example there are five DAQ lists all with a size of 10 (0x0A).

Setup DAQ list for each DAQ. CANape will send a sequence of SET_DAQ_PTR (0x15) and WRITE_DAQ (0x16) commands to setup the data in each DAQ list.
DAQ List Number 0 with ODT number 0 and one element in the ODT (0). This is for measurement variable map1InputY.

```
667           Rx   d 8 15 72 00 00 00 07 00 00
7E1           Rx   d 8 FF 00 72 00 32 D1 FF FF
667           Rx   d 8 16 73 01 00 0D 4A 04 00
7E1           Rx   d 8 FF 00 73 00 32 D1 FF FF
```

DAQ List Number 1 with ODT number 0 and one element in the ODT (0). This is for measurement variable map1Output.

```
667           Rx   d 8 15 74 01 00 00 4A 04 00
7E1           Rx   d 8 FF 00 74 00 32 D1 FF FF
667           Rx   d 8 16 75 01 00 0E 4A 04 00
7E1           Rx   d 8 FF 00 75 00 32 D1 FF FF
```

DAQ List Number 2 with ODT number 0 and one element in the ODT (0). This is for measurement variable map1InputX.

```
667           Rx   d 8 15 76 02 00 00 4A 04 00
7E1           Rx   d 8 FF 00 76 00 32 D1 FF FF
667           Rx   d 8 16 77 01 00 0C 4A 04 00
7E1           Rx   d 8 FF 00 77 00 32 D1 FF FF
```

CANape sends the Start/Stop command sequence to the ECU. In the device configuration window in CANape, the 10 ms Task is using channel 0xA and the 100 ms Task is using channel 0xB.
CANape sends the START_STOP command with mode 0x02 to inform ECU to prepare to start transmission for DAQ list 0 on channel 0xA.

```
667           Rx   d 8 06 78 02 00 00 0A 01 00
7E1           Rx   d 8 FF 00 78 00 32 D1 FF FF
```

CANape sends the START_STOP command with mode 0x02 to inform ECU to prepare to start transmission for DAQ list 1 on channel 0xB.

```
        667             Rx   d 8 06 79 02 01 00 0B 01 00
        7E1             Rx   d 8 FF 00 79 00 32 D1 FF FF
```

CANape sends the START_STOP command with mode 0x02 to inform ECU to prepare to start transmission for DAQ list 2 on channel 0xA with a transmission prescaler rate of two. This is for the 20 ms cyclic event.

```
        667             Rx   d 8 06 7A 02 02 00 0A 02 00
        7E1             Rx   d 8 FF 00 7A 00 32 D1 FF FF
```

CANape sends the START_STOP_ALL command with mode 0x01 to inform ECU to start the transmission of the DAQ data.

```
        667             Rx   d 8 08 7B 01 00 00 00 00 00
        7E1             Rx   d 8 FF 00 7B 00 32 D1 FF FF
```

The ECU will then send the DAQ-DTO messages at the required intervals.
PID number 0x14 is sent every 20 ms.

```
        7E1             Rx   d 8 14 06 45 00 A8 FF E0 00
```

PID number 0x00 is sent every 10 ms.

```
        7E1             Rx   d 8 00 05 45 00 A8 FF E0 00
```

PID number 0x0A is sent every 100 ms.

```
        7E1             Rx   d 8 0A 08 45 00 A8 FF E0 00
```

Note the PID numbers are determined by the size of the DAQ. Each DAQ in this example had a size of ten. Therefore DAQ list zero would use identifiers 0-9, DAQ list one would use identifiers 10-19, and DAQ list two would use identifiers 20-29. Note the CCP limitation of a maximum of 255 available PIDs.

Finally, when the data measurement is stopped, CANape sends the stop command.
CANape sends the START_STOP_ALL command with mode 0x0 to inform ECU to prepare to stop transmission of the DAQ data.

```
        667             Rx   d 8 08 7C 00 00 00 00 00 00
        7E1             Rx   d 8 FF 00 7C 00 32 D1 FF FF
```

Note that the START_STOP_ALL command that was new to the CCP v2.1 specification. The CCP implementation v2.0 did not support the STOP_START_ALL command. CANape can be configured to work with CCP versions 1.01, 2.0, and 2.1.

3.0 Important Details, Special Constraints, or Other Influences

The Device Configuration settings in CANape and the database address of calibration values influence the types of CCP command sequences that are generated by CANape.

4.0 Additional Resources

The following material may provide further information of use to your endeavor.

CCP CAN Calibration Protocol
ASAP Standard
Version 2.1
February 1999

Also, application note:
AN-AMC-1-102 Introduction to the CAN Calibration Protocol

5.0 Contacts

Vector Informatik GmbH
Ingersheimer Straße 24
70499 Stuttgart
Germany
Tel.: +49 711-80670-0
Fax: +49 711-80670-111
Email: info@vector-informatik.de

Vector CANtech, Inc.
39500 Orchard Hill Pl., Ste 550
Novi, MI 48375
Tel: (248) 449-9290
Fax: (248) 449-9704
Email: info@vector-cantech.com

VecScan AB
Fabriksgatan 7
412 50 Göteborg
Sweden
Tel: +46 (0)31 83 40 80
Fax: +46 (0)31 83 40 99
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