Leader in Photon Counting

Manual For The LabVIEW™ DCC .dll Wrappers

Written by

Frank Quadt,M.sc.

February 11, 2021

Contents

Preface

Preface

Thank you for working with Becker&Hickl's DCC and DCU modules.
You are already familiar with B&H's own control software Spcm.exe. There you have the full control over all different module types and their operation modes.
Nevertheless, you feel that there is an important feature missing to bring your application to its expected heights.
B&H honours that and is offering you software drivers capable to control your DCC and DCU module down to the lowest level.
These drivers are written in National Instruments LabWindows / CVI and available on your computer at C:\Program Files (x86)\BH\DCC\DLL\dccXX.dll (XX standing for your operating system's bitness).
For programmer wanting to use National Instruments LabVIEW to control their DCC and DCU modules, B&H provides LV wrappers in the Becker&Hickl DCC Modules folder to call the above mentioned dll.
Also included are some higher-level convenience VIs as well as some simple examples to provide you with a starting point to your own LV-project.

This manual is about the use of these wrappers and examples, aimed at users with a solid understanding of programming technics in LabVIEW.
For training in LabVIEW itself we recommend to contact National Instruments directly at www.ni.com.

Attention: Notice that the LabVIEW Plug and Play Drivers are just usable with the corresponding DLL. Otherwise these Drivers will fail to load!

1 Installation of the LabVIEW™ DCC .dll Wrappers

The Becker&Hickl DCC Modules LabVIEW project comes as collection of files in a folders hierarchy installed together with your Spcm.exe software at C:\Program Files (x86)\BH\DCC\BH-LabVIEW-DCC.
The VIs and the project are written in the LabVIEW2017 development environment. Upgrading to a newer development environment on your PC should not be a problem. For downgrading to older versions use the File>Save for Previous Version… option in the LV project window or get in contact with Becker&Hickl.

All LV-wrapper VIs in the \DCC folder depend on the dccXX.dll located at C:\Program Files (x86)\BH\DCC\DLL on your computer. There are 3 versions available. You will need to use dcc32.dll on Windows32 bit systems. For Windows64 bit systems you will use the dcc32x64.dll when programming in the LabVIEW 32bit environment or dcc64.dll when running LabVIEW 64bit.

We recommend to always upgrade your Spcm.exe to the newest version which will include new features and bug fixes.
With every new release there will be new versions of the dccXX.dll. Since B&H reserves the right to change (with release notes) functions and their parameter sets at any time, we also recommend not to edit the wrapper VIs directly but keep them in their original form at their original location. Like this an upgrade of Spcm can overwrite the wrapper VIs without changing your code.

To start your own LabVIEW project, you should create a folder on your User\My Documents path, create a New Project on the LabVIEW start-up window. In the project window right click on My computer and choose Add>Folder (Auto-populating)… In the file select box navigate to C:\Program Files (x86)\BH\DCC\BH-LabVIEW-DCC\ and click Select Folder.
You could do the same for the Examples folder, but we recommend to create a copy of this folder on your development path. You most likely will want to do changes on these VIs which you might not be allowed to on the protected path of C:\Program File (x86). They also might be overwritten with newer Spcm.exe version distribution.

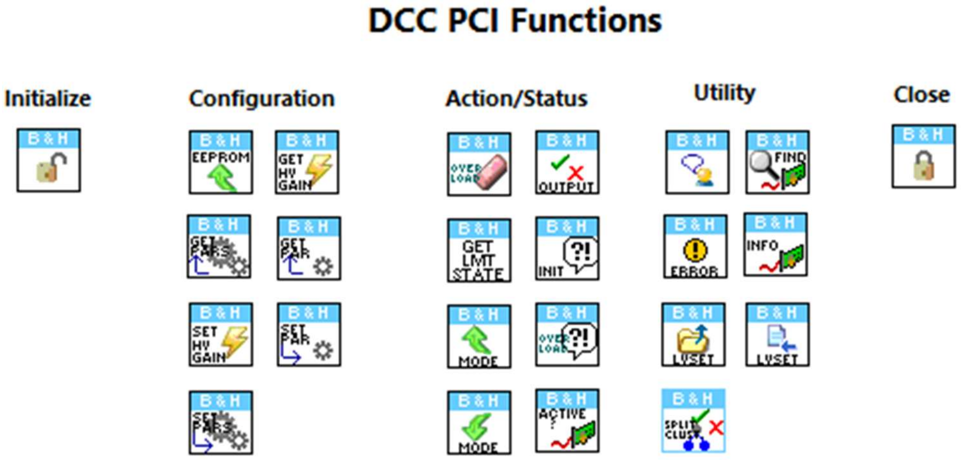
When attempting to write a DCC control application we suggest to start off with one of our examples located in the \Examples folder.

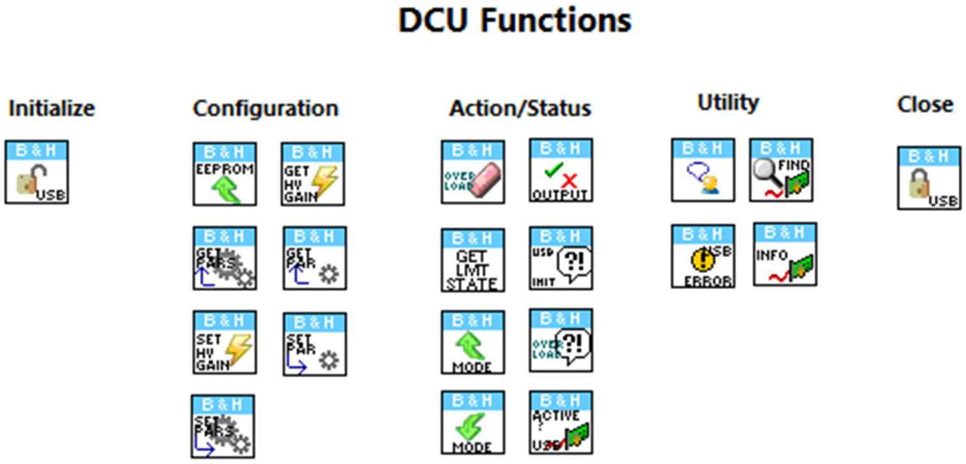
2 The DCC and DCU Module Functions

The DCC/ DCU module functions provide a full range of procedures for controlling the module(s) totally. Besides the possibiliy to initialise and close the module properly a variety of functions for controlling the different outputs will be provided.

Further, since 2019 Becker & Hickl provide additionally to the PCI and PCI express cards also a USB version of the DCC module named DCU. These functions are also included inside the DCC DLL. So one has the full flexibility to choose the device which fits best.

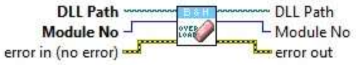
In the following just the functions for the DCC will be described. However, the upcoming functions have the same behaviour for the DCU but for a better overview these functions are split into two parts in the folder tree.





2.1 Action–Status Functions

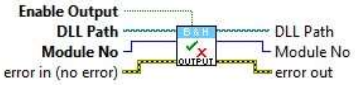
2.1.1 The Clear Overload Function



The clear overload function clears overload hardware flags on the specified DCC module. After that it will enable again the outputs which were disabled by overload.

Caution: Please stay alert that the function may control an external high voltage power supply or a detector that can be damaged by exceeding the maximum operation voltage or the maximum output current. In particular, if you control a high voltage power supply, make sure that it is safe to turn on the voltage.

2.1.2 The Enable Outputs Function

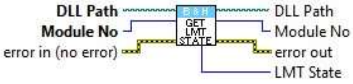


The enable outputs function is used to enable or disable the outputs of the DCC module which is specified by the module number. Notice that the outputs should be disabled during connecting cables or setting up devices controlled with DCC module(s).

Caution: Please stay alert that the function may control an external high voltage power supply or a detector that can be damaged by exceeding the maximum operation voltage or the maximum output current.

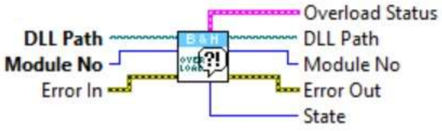
In particular, if you control a high voltage power supply, make sure that it is safe to turn on the voltage. Switching off a high voltage power supply by this function is **not safe** in terms of human safety, and disabling the outputs cannot be used to exclude hazard by the output voltage of externally connected high voltage power supply.

2.1.3 The Get Current LMT State Function



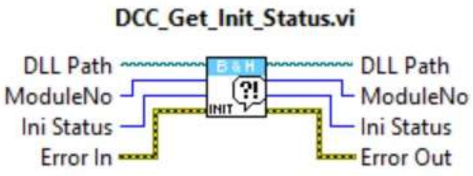
This function is used to check whether the current limit is reached (for cooling device on connector 3) on the DCC module specified by module number.

2.1.4 The Get Overload State Function



The procedure is used to check the current overload status occurred on the specified DCC module. It checks bitwise the current status and shows in boolean cluster indicator whether an overload occurred.

2.1.5 The Get Init Status Function



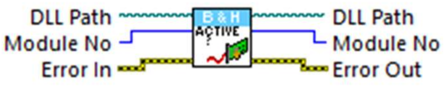
This function is a low-level procedure and should be called directly after the “Initialize.vi”. It loads the state variable with the initialization resulted code.

Possible values are shown below.

Initialisation Code	Description
0	No Error
−1	Init Not Done
−2	Wrong EEPROM Checksum
−3	Cannot Open Device
−4	Module Already in Use
−5	Corrupeted License Key
−6	License Key Not Read from Registry
−7	License is Not Valid for DCC DLL
−8	License Date Expired

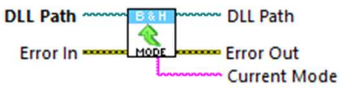
2.1.6 The Test If Active Function

The “Test If Active Function” returns information whether the DCC module defined by “Module No.” is active or not, which are represented as return values 1 or 0 respectively. As a result of a wrong initialisation a module can be deactivated. To find out the reason of deactivating the module, run the “Get Init Status Function”.

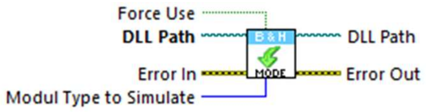


2.1.7 The Get/Set Mode Function

The DLL can handle two different modes. The hardware and the simulation mode respectively. In which mode the DLL currently operates can be read out by the use of the get mode function.



The set mode function can change the currently set operation mode of the DLL to one of the modes. Further, the set mode function can force to use the non–active module.

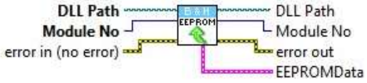


This can be necessary i.e. after a crash of the application.

2.2 Configure Functions

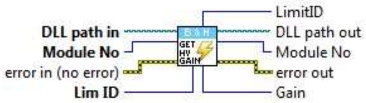
2.2.1 The Get EEPROM Data Function

The cluster ”EEP Data” is filled with the contents of the EEPROM of the DCC module specified by the module number. The EEPROM contains the production data of the module. The cluster ”DCC EEP _Data” is defined in the file ”dcc_def.h”.



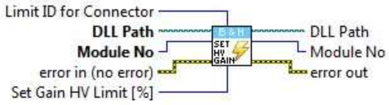
2.2.2 The Get Gain HV Limit Function

The procedure loads “Value” with the actual value of the gain HV limit from the EEPROM of the DCC module specified by the module number. Depending on the limit ID connector 1 or connector 3 gain HV will be used. Gain HV limits are expressed in percent of the maximum voltage which can be sent to gain HV outputs.



2.2.3 The Set Gain HV Limit Function

The procedure sets the gain high voltage (HV) limit of the DCC module to the value taken from the given parameter value. Depending on the specified limit ID (0 for connector 1, 1 for connector 3) connector 1 or connector 3 will be used.



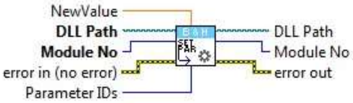
The limits are stored in the module EEPROM of the DCC module. Then the corresponding parameter, namely ’C1 GAIN HV’ or ’C3 GAIN _ HV’ will be recalculating according to the limit.

On return the ’value’ variable is set to the current limit value read from EEPROM. In case of errors during writing or reading to/from DCC EEPROM gain _HV limit is set to the default value of 100. Therefore, please make sure that no error occurred and use the function ’DCC get gain HV limit’ to check that the correct limit is set. ’Gain _HV limits’ are expressed in % of the maximum voltage which can be sent to gain HV outputs.

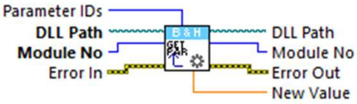
Caution: This function limits the detector operation voltage essentially by software. It gives reasonable safety against unintentional overload of detectors or other connected devices. The function is not safe in terms of human safety, and it cannot be used to exclude hazard by the output voltage of externally connected high voltage power supplies. Please make sure that a connected high voltage power supply, the devices to which the high voltage is connected, or other devices connected to the DCC are safe for the full output voltage range.

2.2.4 The Get/Set Single Parameter Function

The procedure loads the requested parameter from the

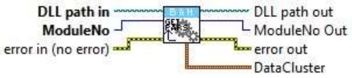


just the current value of

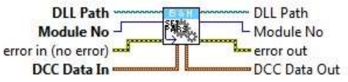


internal DLL data structures of the DCC module specified by the module number. Notice that the first two parameter ID values (0 – ”ACTIVE” and 1 –” PCI _ BUS NO”) are not allowed and will return an error code. The parameter ID values are defined in ”dcc_def.h” file as ”DCC PARAMETERS KEYWORDS”.

2.2.5 The Get/Set Multi Parameters Function



The “Set Multi Parameters Function” sets **all of the specified hardware parameters**. This procedure sends **all parameters** from the ”DC- Cdata” cluster to the internal DLL structures structures.



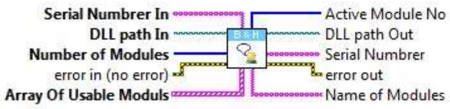
The new parameter values are recalculated according to the parameter limits and hardware restrictions. Furthermore, cross dependencies between different parameters are taken into account to ensure the correct hardware operation. It is recommended to read back the parameters after setting to get their true values after recalculation. The values of ”base adr” and ”active” are not changed. They can be changed only by a new INI file and a new DCC init call.

If an error occurs for a particular parameter, the procedure does not set the rest of the parameters and returns with an error code.

The “Get Multi Parameters Function” reads **all of the set parameters** which were present in the module after calling the “DCC Init Function” or the “Set Multi Parameters Function” was called. This procedure transfers **all of the parameter values** of the DCC module specified by module number from the internal DLL structures into a cluster of the type “DCCdata”. The parameter values in this structures are described in ”DCCDLL.doc”.

2.3 Utility Functions

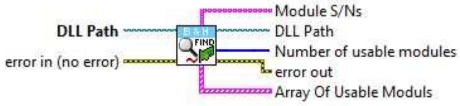
2.3.1 The Ask User For Modules Function



This function was introduced to give the user the possibility to scan the system after usable modules. These modules will then be listed and the user can choose which one will be used in the running application. The return value is a numerical I16 array which contains the module number of the chosen modules. Due to the fact that the system will be scanned this function is not intend to use in parallel.

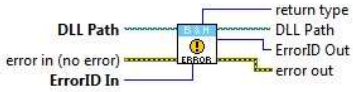
2.3.2 The Find Modules in Use Function

The ”DCC_find_modules_in_use.vi” checks which measurement card is currently in use. These which are not in use will be displayed in the ”Array Of Usable Modules”



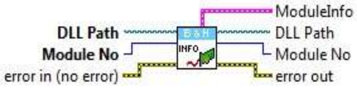
2.3.3 The Get Error String Function

The ”DCC_get_error_string.vi” get the corresponding error string in dependence of the given error ID. For that the given return value from the previous executed function will be evaluated.



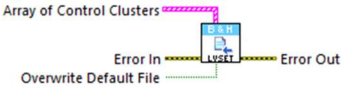
2.3.4 The Get Module Information Function

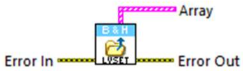
After calling the DCC initialisation function the internal ’DCCModInfo’ structures for up to 8 modules are filled. This function transfers the contents of the internal structure of the DLL into a structure of the type ’DCCModInfo’ (see ”dcc_def.h”). The ’DCCModInfo’ structure contains information like the currently used module type, bus number and slot number as well as the base I/O address and the module serial number.



2.3.5 The Save/Load to LVSET Function

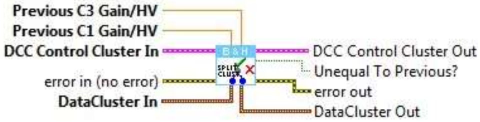
This function stores the DCC control cluster to a *.lvset file. The user can choose between two options. Store and write a new file or overwrite the default settings. The default settings are located inside the BHdata folder. By using the default installation folders, the corresponding BHdata folder is located on the system partition C.





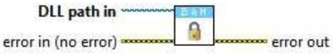
2.3.6 The Split Cluster and Check Function

Splits the corresponding cluster to its elements and checks whether there had occurred a change in values. Further, just these values will be handled which were changed. If there is a change all current values will be written into the data cluster of the DCC module.

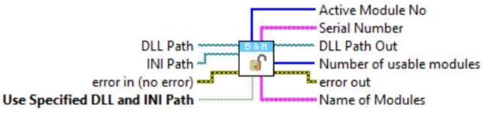


2.4 The Close Function

This closing function closes all open references and closes the DLL reference in a proper way and brings the internal DLL structure in a state before the initialising function was called.



2.5 The Initialise Function



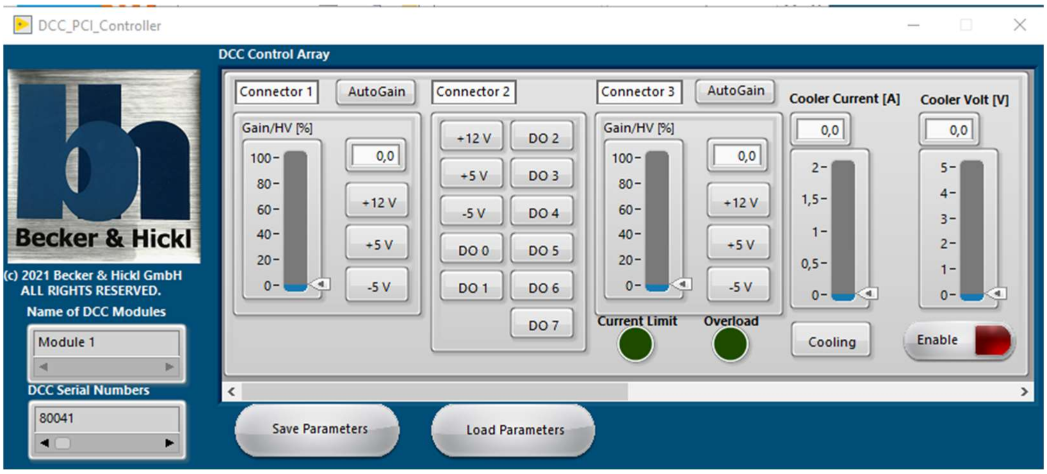
Establishes communication with the cards installed in the computer system or with an external device which includes DCC cards. It also checks how many cards are available and ask the user which cards are going to be used. Further, it prepares the cards for further driver operations. Therefore, call this VI before calling other driver VIs for this device. Generally, you need to call the **Initialize** VI only once at the beginning of an application.

2.6 The Example Applications

2.6.1 Simple Measurement VI

Together with the drivers Becker & Hickl GmbH provides also a simple measurement application. This application shows the user how to use specific functions which were described earlier on.

Furthermore, this application, which was created also in LabVIEW™, provides the possibility to control the DCC (PCI) completely. Moreover, this application was intended to work with multiple DCC modules at the same time but can also handle just one DCC module alone.


Name of DCC Modules: Module 1

DCC Serial Numbers: 80041

Figure 10: The front panel of the simple measurement VI for controlling the DCC (PCI) module. Up to eight different modules can be handled. Notice the possibility to change the different connectors separately for each used DCC module.

As shown in Fig.10 an array structure was used to control every DCC module individually. Notice, that per module every connector can be controlled individually. However, make sure that all connected devices can handle the correct power to prevent damage on the DCC card as well as the connected device(s). If a detector is connected to connector 1 or connector 3 and an overload occurred this will be indicated separately per module via a LED on the bottom right-side of the application by turning red. If so, the output will be disabled automatically.

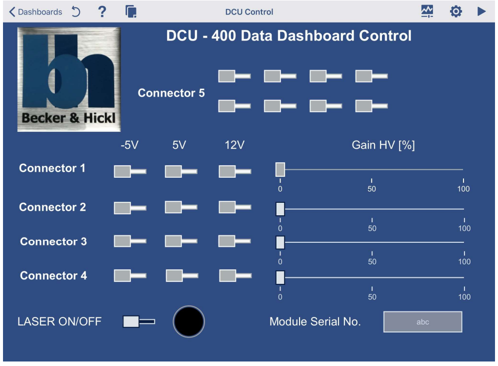
Further, it is possible to save and load the current set of DCC settings to make it possible to start where you left your system the last time. Notice, that the ”Enable Output” button is disabled when settings are loaded.

2.6.2 Using the Data Dashboard

NI™ provides an application for remote access to a self-written LabVIEW™ application. This application, namely "Data Dashboard", has the possibility to send and receive data by using so–called "network–published shared variables" and with deployed LabVIEW Web services on i.e., "indicators" which can be implemented into your source code easily. The Data Dashboard™ application supports iOS, Windows as well as Android operation systems and can be downloaded from the corresponding App Store free of charge.

Figure 11: The Data Dashboard™ logo from NI™ company.

For that, the Data Dashboard brings the possibility to use your application over the internet. Becker & Hickl GmbH provides for that a ready to use Data Dashboard front panel which can be downloaded from our website and is part of the Becker & Hickl LabVIEW™ Plug and Play Drivers. It is pre–configured and suitable for the shipped examples. For further information please take a look onto www.ni.com by using the keywords "Get Started with Data Dashboard for LabVIEW™". For using the provides Data Dashboard LVDD file just four simple steps are necessary:



1. Save the LVDD file onto your hard disc/ tablet.
2. Click and hold to open menue
3. Click "Share"
4. A small number of app's can be chosen now. Please choose "Copy to Dashboard".

Now the NI Data Dashboard will be opened and saved for future use.

Known Issues

In the following section frequently ask questions as well as known issues will be listed. If you here do not find a solution to your specific problem please feel free to get in touch with Becker & Hickl GmbH.

LabVIEW do not find the previous copied drivers If LabVIEW asks where to find the different VIs for your specific measurement card please make sure that you had started the correct LabVIEW version. Maybe you started a 32–bit version instead of the 64–bit version.

LabVIEW crashes during the use of the LabVIEW Plug & Play Drivers
This issue can happen, if the corresponding used DLL has a wrong version. There exists a minimum DLL version for each LabVIEW Plug and Play driver. In the following the minimum DLL version number for each Becker & Hickl LabVIEW Plug and Play driver will be provided.

Package	Version
DCC	2.5
DDG	3.0
GVD	1.9
PMS	3.4
SPC	4.6

Contact Information

Becker & Hickl GmbH provide a huge amount of possibilities to establish your experimental setup. You need help, support or you want to discuss some specific topics or issues so please get in touch with us.

Becker & Hickl GmbH
Nunsdorfer Ring 7 – 9
12277 Berlin Germany

Phone: +49(30)2128002 – 0
Fax: +49(30)2128002 – 13

Email: info@becker–hickl.com

If you have some specific questions, annotations or criticism referring the programming of the LabVIEW™ DCC .dll wrappers, please get directly in touch with the programmer:

Becker & Hickl GmbH
To The Attention To:

Frank Quadt, M. sc

Nunsdorfer Ring 7 – 9
12277 Berlin Germany

Phone: +49(30)2128002 – 0
Fax: +49(30)2128002 – 13

Email: quadt@becker–hickl.com

Software Graphic & Design: Dr.C. Junghans and F. Quadt
Text/Layout: Dr.C. Junghans and F. Quadt
Developer: F. Quadt