

USER MANUAL

COMPACTRIO RESOLVER TO DIGITAL CONVERSION MODULE RDK9316

Art.Nr. 86257



SET GmbH
August-Braun-Str. 1
88239 Wangen/Allgäu
Germany
Tel.: +49 (0)7522 91687-600
Fax: +49 (0)7522 91687-899

Copyrights:

SET GmbH
All rights reserved.

Information contained herein is the property of SET GmbH and shall not be duplicated, copied, used or disclosed in whole or in part of any purpose. The right of duplication, use or disclosure is permitted only by written agreement of SET GmbH, August Braun Str. 1, 88239 Wangen/Germany.

LabVIEW, CompactRIO, TestStand are Trademarks of National Instruments

Table Of Contents

1.	Important Instructions	4
1.1	Initial Inspection	4
1.2	Safety Instructions.....	4
1.2.1	Module Failure	4
1.2.2	Impermissible Applications	4
1.2.3	Module Installation And Removal	4
1.2.4	Electrical Connections	4
2.	Module Overview	5
3.	Getting Started	6
3.1	Software-Requirements	6
3.2	Driver-Installation	6
3.3	RDK9316 Compact RIO LabVIEW Drivers	7
3.3.1	Copying The Driver Files Into New Projects	11
3.3.2	NI PCI/PXI FPGA-Card Driver Application.....	13
3.3.2.1	Creation of a new Project	13
3.3.2.2	Creating a New FPGA-Target	14
3.3.2.3	Adding An R-Series Expansion Chassis	16
3.3.2.4	Adding The RDK9316 Driver Components To A Project.....	17
3.3.2.5	Adding RDK9316 Modules To The Project Via Drag-And-Drop Function	18
3.3.2.6	Adding RDK9316 Modules To The Project Via The Discovery-Function	19
3.3.2.7	Adding FPGA Example-Applications.....	23
3.3.2.8	Saving The Project	24
3.3.2.9	Compiling And Running The Example Application	25
3.3.3	Using The Driver Together With A NI RealTime CompactRIO	26
3.4	Connecting The Resolver	27
3.5	Running The Application Example	27
4.	Application Development.....	28
4.1.1	RDK9316 Driver And Communication	28
4.1.1.1	Driver-VI Implementation.....	28
4.1.1.2	Triggering RDK9316 Functions Via Functions-VIs	29
4.1.1.3	Initialising And Clearing The FIFO Buffers	29
4.1.1.4	Application To Module Communication Diagram	31
4.2	RDK9316 Driver-Instructions	33
4.3	RDK9316 Driver Error Codes	37



4.4	Module Identification Under LabVIEW	39
4.5	Saving The Setup	39
4.6	Auto Ratio	39
5.	Technical Specification	40
5.1	Housing	40
5.2	External Power Supply.....	40
5.3	Excitation Output	40
5.4	Sinus And Cosine Signal Inputs	41
5.5	Position Processing.....	41
5.6	Environmental Conditions	41
5.7	Connector Pinout.....	42
6.	Module Calibration.....	42
7.	Module Maintenance	42
8.	Service Address	42

1. IMPORTANT INSTRUCTIONS

Please read all instructions carefully before the RDK9316 is installed into a cRIO chassis or connected to other equipment!

1.1 INITIAL INSPECTION

Check that the shipment is complete and note whether any damage has occurred during transport. If the contents are incomplete or there is damage, file a claim with the carrier immediately, and notify the SET GmbH service contact to facilitate the repair or replacement of the module. The address is listed in the back of the manual.

The following parts should be included in the shipment:

- ✓ RDK9316 module
- ✓ CD-ROM with driver software, application examples and manual

1.2 SAFETY INSTRUCTIONS

1.2.1 MODULE FAILURE



Do not install the RDK9316 module into a cRIO chassis when the module is obviously damaged:

- physical damage
- lose parts inside the module

1.2.2 IMPERMISSIBLE APPLICATIONS



The module is designed for laboratory use. Installing or operating the module in explosive or hazardous environments is not permissible and may result in serious injury or death!

1.2.3 MODULE INSTALLATION AND REMOVAL



Hot Surface!

The module may be hot. Touching the module may result in body injury!

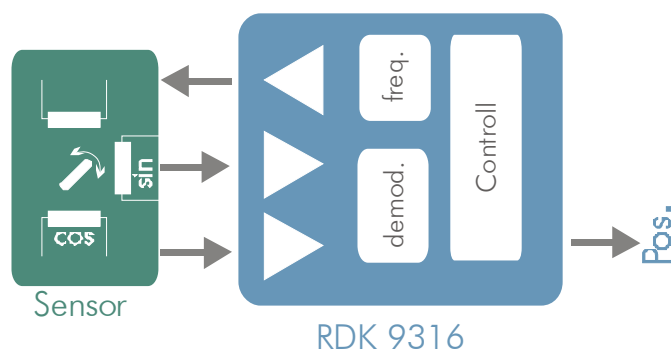
1.2.4 ELECTRICAL CONNECTIONS



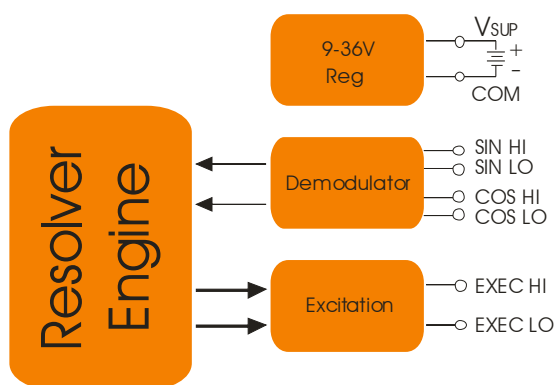
The module is not designed to isolate voltage levels of more than 50V_{DC}. Do not exceed the voltage levels according to the technical specifications. Not following this instruction may result in module damage and serious injury or death!

2. MODULE OVERVIEW

RDK 9316 is a compact, reliable and highly versatile Resolver-to-digital conversion module. All commonly used Resolvers can be easily connected to the module without any additional signal adaption. The module has a built-in excitation oscillator inclusive power stage which can drive most Resolvers directly without the necessity of an external power booster. All module parameter are interactively adjustable via software. A sensor-detection functionality automatically adjusts the module to the specific sensor ratio.



The RDK 9316 accepts a wide voltage supply range from 9V to 36V and provides galvanic isolation between the cRIO rack and the demodulator interface. The module is operable within the NI cRIO real time environment and can also be plugged into a NI R-series expansion chassis with PCI / PXI FPGA card. Included in delivery are all drivers required for the cRIO systems and LabVIEW integration examples.



RDK9316 applications include industrial and military position control systems such as motor control, robotics and many kinds of servo loops which use Resolvers.

3. GETTING STARTED

3.1 SOFTWARE-REQUIREMENTS

To use the RDK FPGA-Driver the following NI Software Components must be installed first:

- LabVIEW
- LabVIEW FPGA Module
- SET RDK-9316 C-Series Module („Setup.exe“ included on the SET RDK Driver-CD)

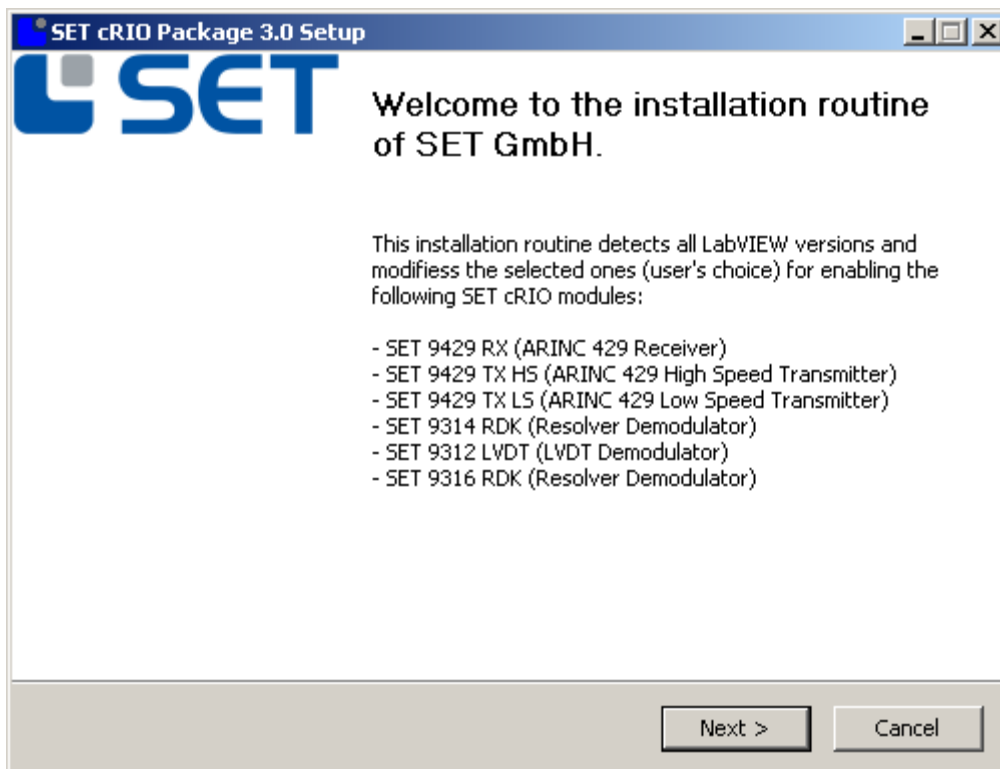
3.2 DRIVER-INSTALLATION

After Installation of the NI Software Components, the RDK driver software itself must be installed. To do this use the RDK9316 CD-ROM and follow the following instructions.

To start the installation process, execute the „setup.exe“ program on the CD-ROM.



Please follow the instructions of the setup program to complete the driver installation.



When the installation is completed successfully, the host system is ready to use RDK9316 modules in a LabVIEW project.

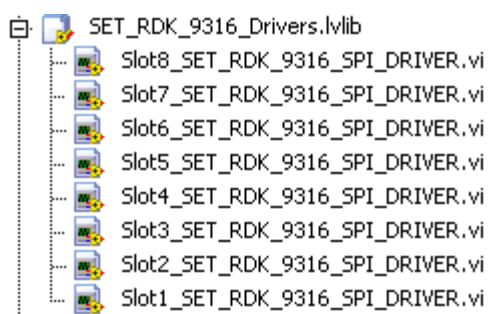
3.3 RDK9316 COMPACT RIO LABVIEW DRIVERS

The driver application is demonstrated within a LabVIEW example project. To copy the driver components into a new project, use the drag & drop function.



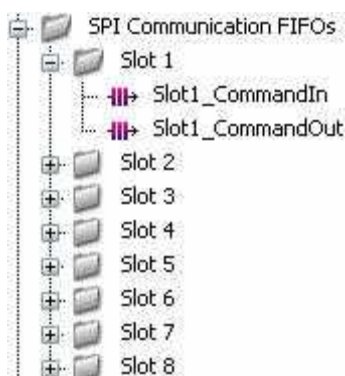
LabVIEW driver components

The LabVIEW FPGA software driver for the RDK9316 module comprises eight driver VI's which control the serial data transfer between the modules and the FPGA via SPI interfaces. (Refer to the VI-library „SET_RDK_9316_Drivers.lvlib“).



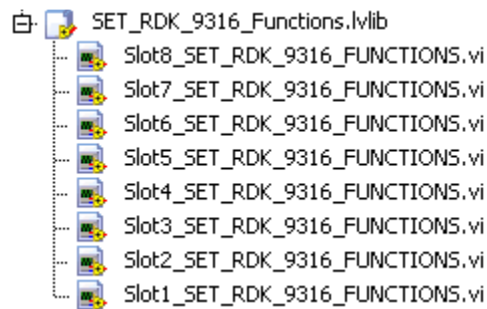
Each module slot applies to its own driver-VI

Each driver VI uses 2 FIFO buffers to communicate with the LabVIEW application. The FIFO buffers are organized in CommandIN-FIFOs and CommandOUT-FIFOs for bidirectional data flow from the application to the module and versus vice. Note that these buffers are pre-defined in terms of name and size and should not be modified to ensure correct operation.



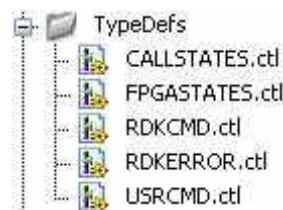
Each driver-VI uses two command FIFOs

The user interface is formed from eight slot specific „Functions-VIs“, which perform the driver calls by use of the communication-FIFOs. (Refer to the VI-library „SET_RDK_9316_Functions.lvlib“).



Each slot applies a specific functions-VI

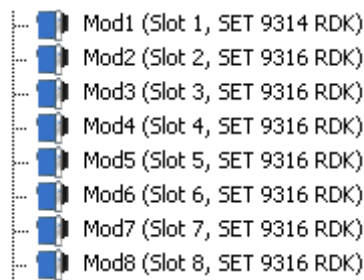
The folder „TypeDefs“ gives data type definitions used within the drivers: error codes and module commands.



LabVIEW driver data type definitions

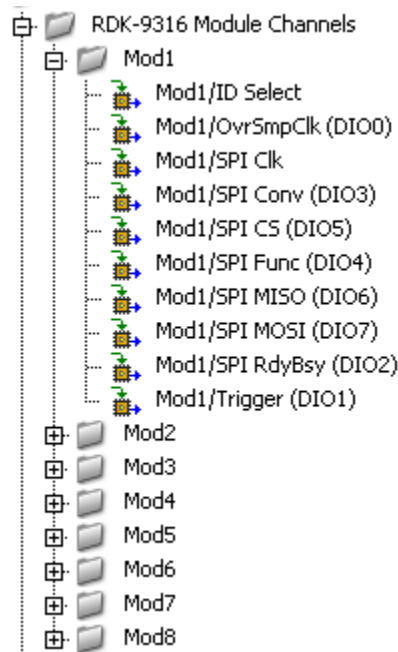
- USRCMD.ctf
Defines the applicable RDK9316 commands
- RDKERROR.ctf
Defines the driver error codes
- RDKCMD.ctf
For driver internal use
- CALLSTATES.ctf
For driver internal use
- FPGASTATES.ctf
For driver internal use

The example project already covers all module slots with the maximum number of modules, which is four, when an R-Series Expansion Chassis is applied and up to eight modules for a cRIO-Chassis. The names of the modules in the project and the correlating modules I/O channels are pre-defined and should not be changed for correct operation.



The example project includes all module slots

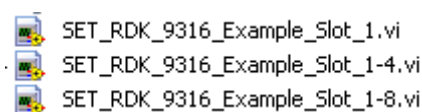
The example project also defines all required data signals:



Example project data signal definitions

The example project modules can be copied to a new project. Alternatively, when the automatic LabVIEW module detection mechanism “Discover C-Series Modules” is used, attention on identifier assignment should be paid to avoid naming conflicts (refer to chapter 3.3.2.6).

Three example files are included to demonstrate the driver-VI application:



Tree example programmes for the R-Series Expansion Chassis and cRIO-Chassis

- SET_RDK_9316_Example_Slot_1.vi

Program example which demonstrates the application of a single RDK9316 module connected to slot 1 of a (PCI) FPGA R-Series Expansion-Chassis or a cRIO-Chassis.

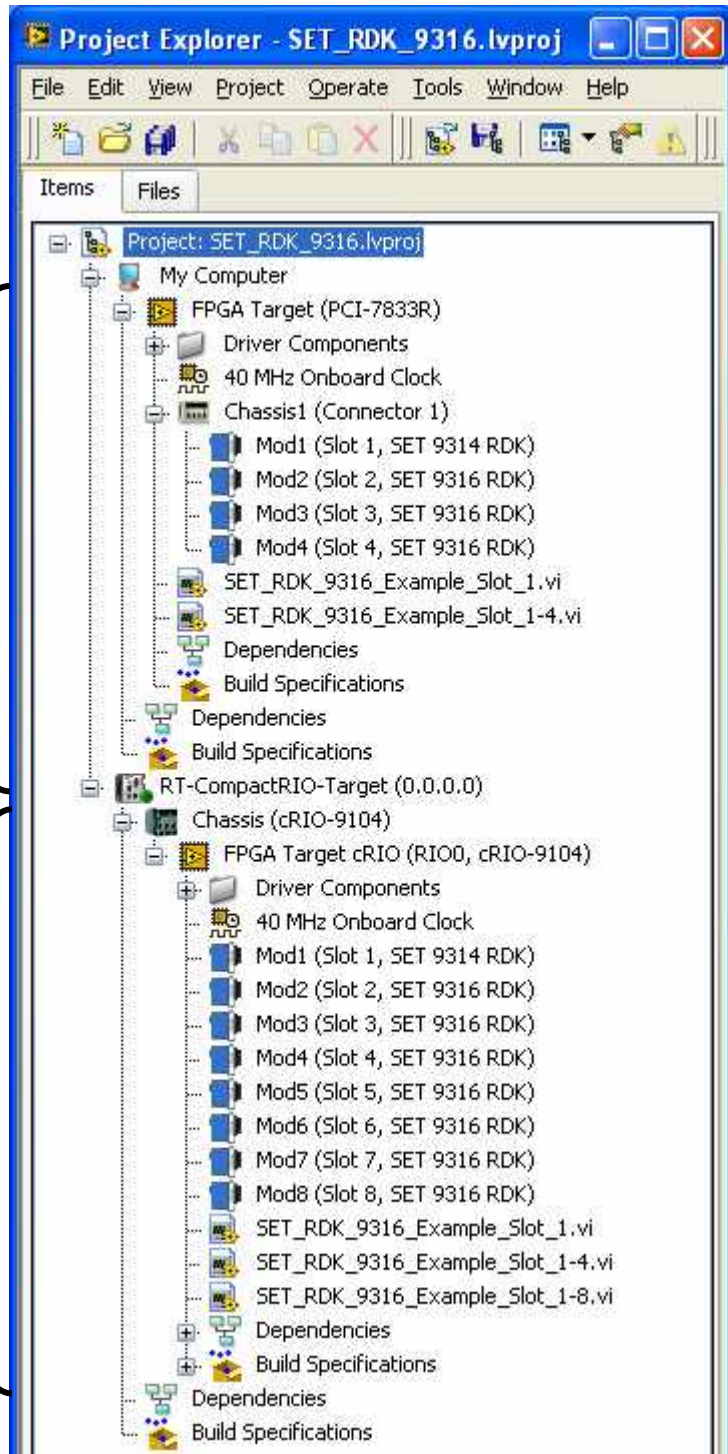
- SET_RDK_9316_Example_Slot_1-4.vi
Program example which demonstrates the application of four RDK9316 modules connected to a (PCI) FPGA R-Series Expansion-Chassis or a cRIO-Chassis.
- SET_RDK_9316_Example_Slot_1-8.vi
Program example which demonstrates the application of eight RDK9316 modules connected to a cRIO-Chassis.

3.3.1 COPYING THE DRIVER FILES INTO NEW PROJECTS

The example below provides all software components to get started with a new project. To copy the files into the new project, simply use the drag&drop function. The example project demonstrates the driver operation on two different target platforms.

LabVIEW driver integration for a NI PCI-FPGA-Card in combination with a (4 Slot) R-Series Expansion Chassis

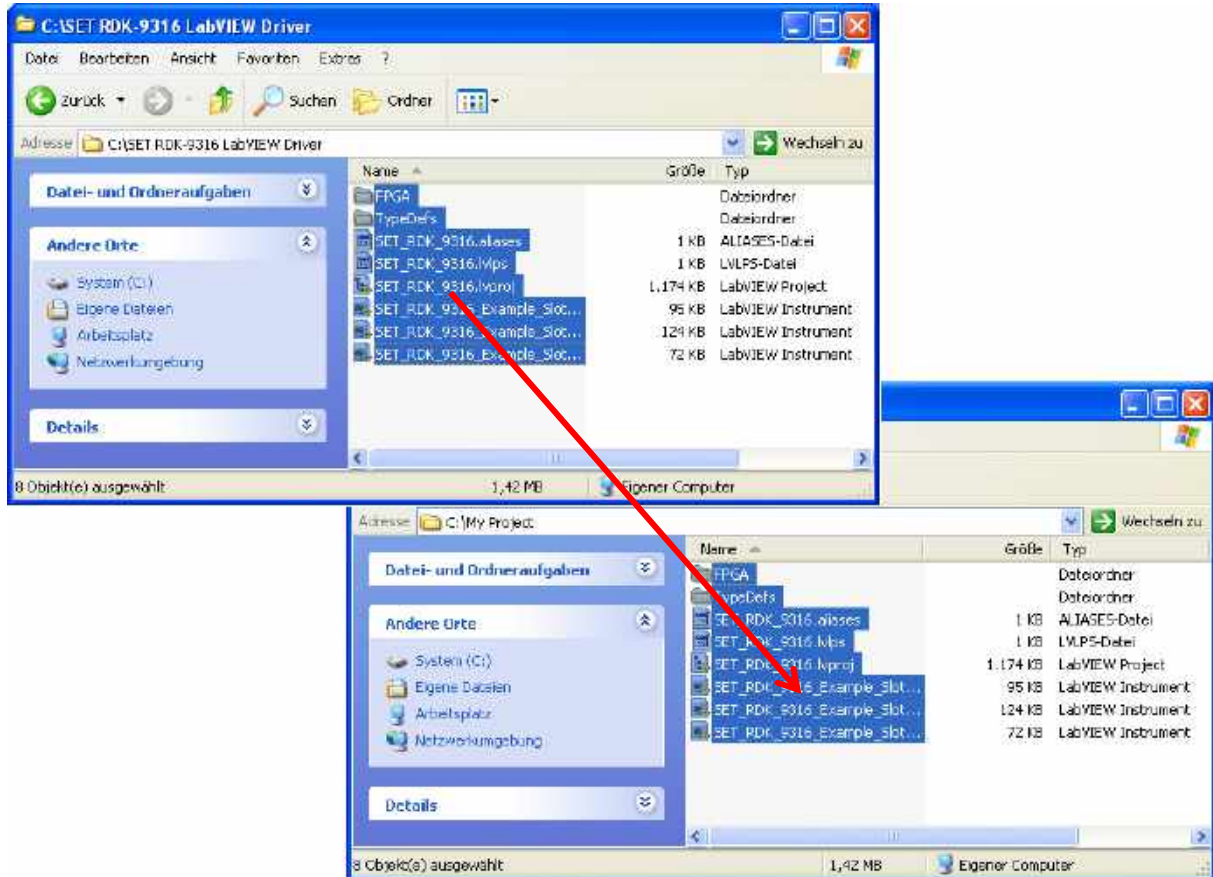
LabVIEW driver integration for a CompactRIO-Chassis with up to 8 slots.



Example Project Structure

To integrate the LabVIEW driver into a new or an existing project, all files from the example project must be copied into the new project folder. Make sure that the file and folder names

are not modified during the copy process as the correct function of the drivers will be affected thereby.



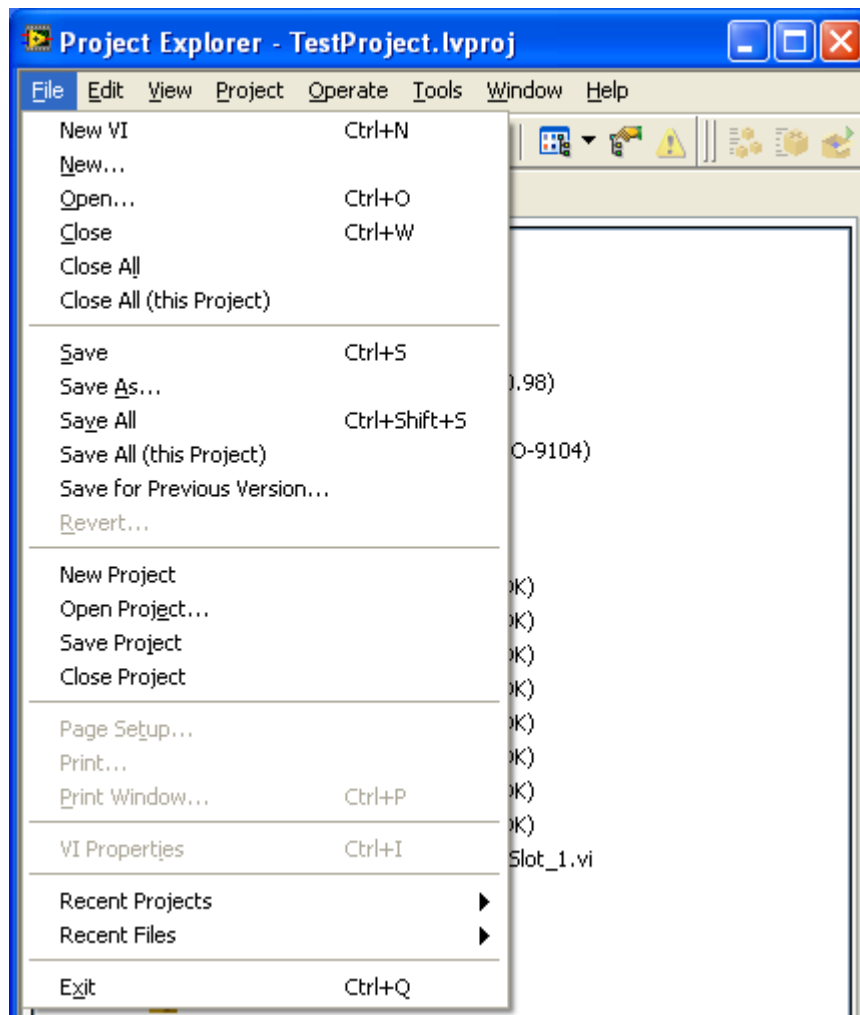
All files must be copied into the new project folder

3.3.2 NI PCI/PXI FPGA-CARD DRIVER APPLICATION

The example below demonstrates the driver integration for a NI PCI-FPGA card together with an R-Series Expansion Chassis.

3.3.2.1 CREATION OF A NEW PROJECT

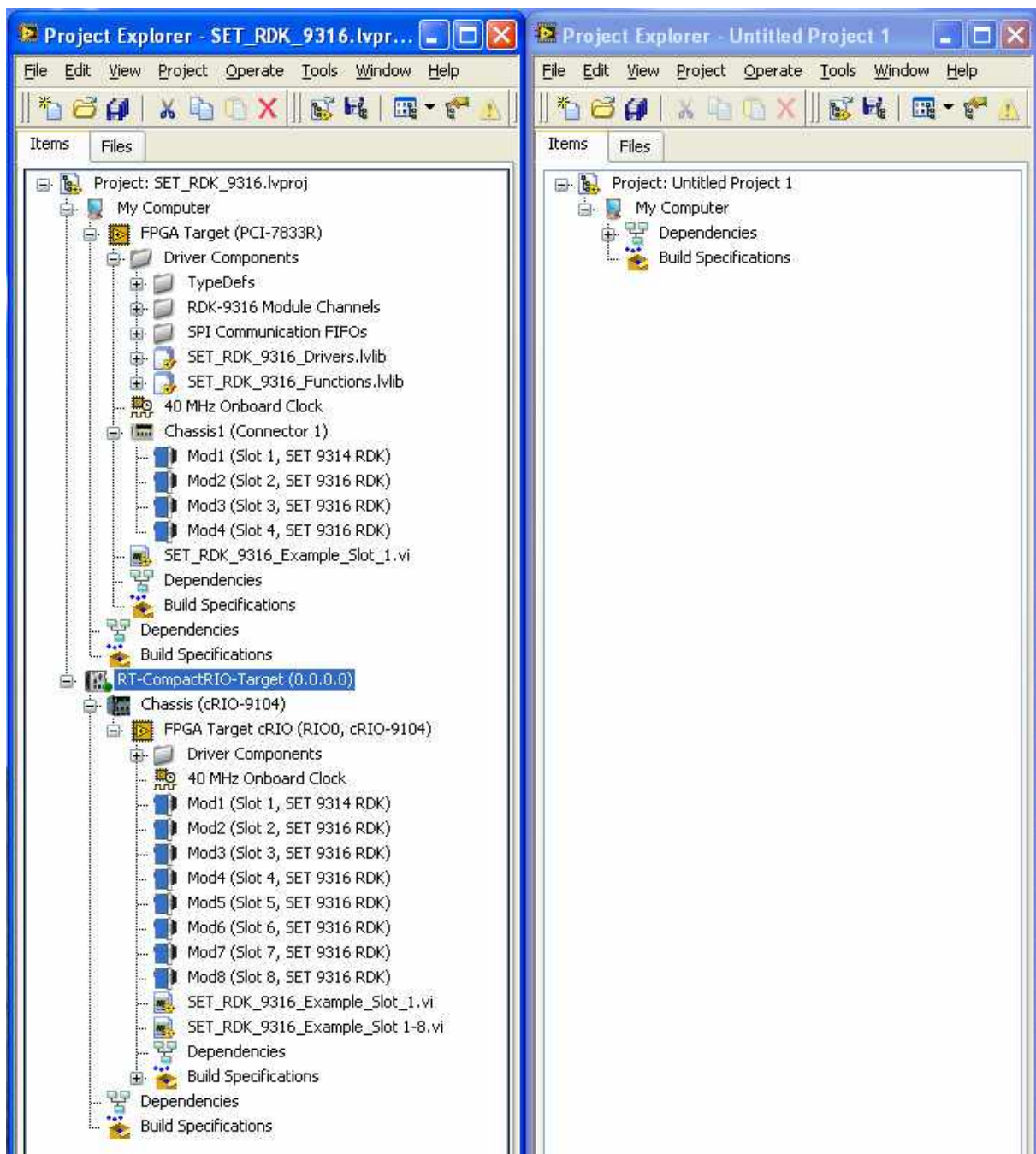
When all driver files are copied into the new project folder, the example project is opened within the new project folder. To integrate the driver into an existing LabVIEW project, the target project must be opened subsequently. Alternatively, to start a new project „File → New Project“ must be clicked.



Opening the target project or creating a new project

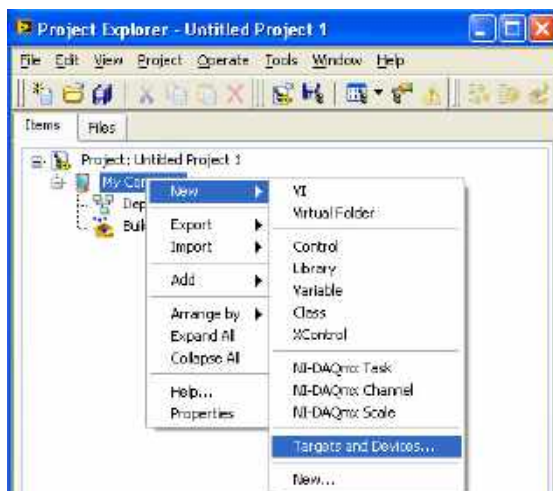
3.3.2.2 CREATING A NEW FPGA-TARGET

Both project windows can be aligned as shown below. The driver components can be copied into a new project via drag-and-drop function. Note that the STRG-key must be pressed during drag-and-drop to copy the element. Otherwise the element will be removed from the example project. To prevent faults, the following module installation sequence must be followed exactly.

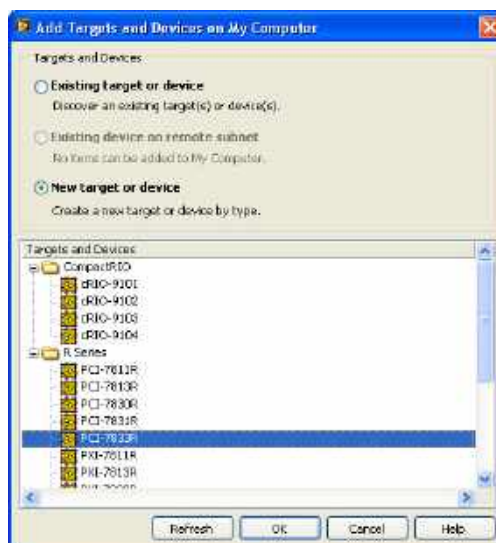


The driver components can be copied by use of the drag-and-drop function

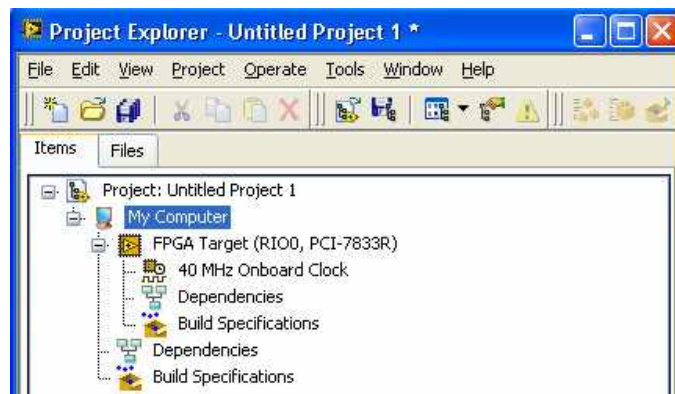
To add a new FPGA-Target use the mouse and click (right button) onto the computer symbol and select: „New → Targets and Devices“.



Select an existing FPGA-Target or define a new device.

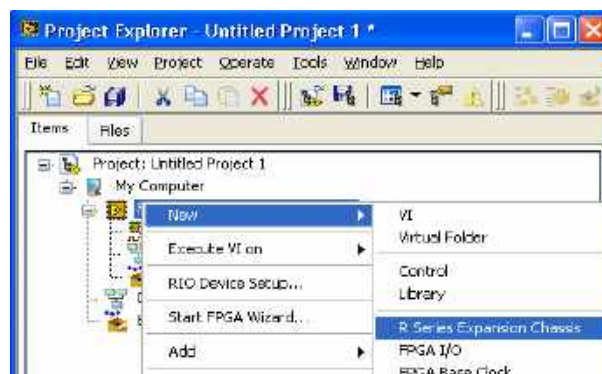


The new FPGA-Target is now added to the project-structure.

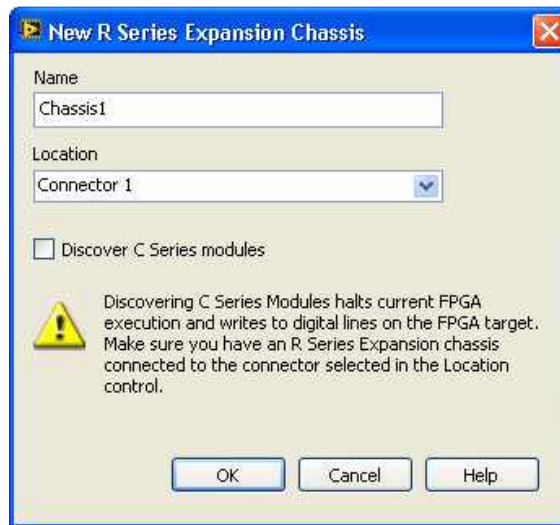


3.3.2.3 ADDING AN R-SERIES EXPANSION CHASSIS

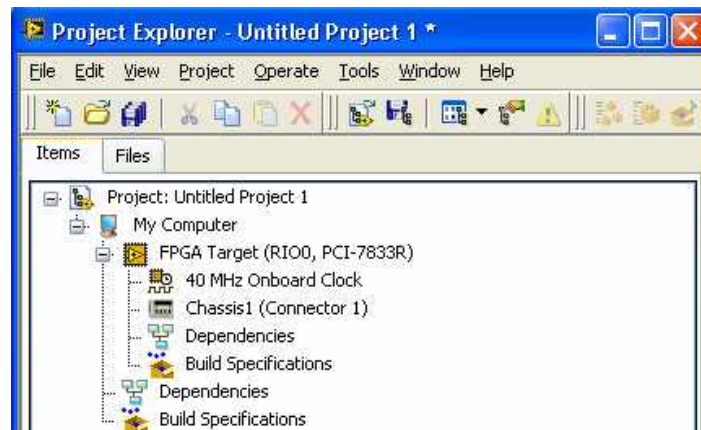
To use an RDk9316 module together with a FPGA chassis, an „R-Series Expansion Chassis“ must be appended to the actual project structure. Use the mouse and click with the right button onto the newly added FPGA-Target and select „New → R Series Expansion Chassis“.



Confirm the dialog with „OK“.

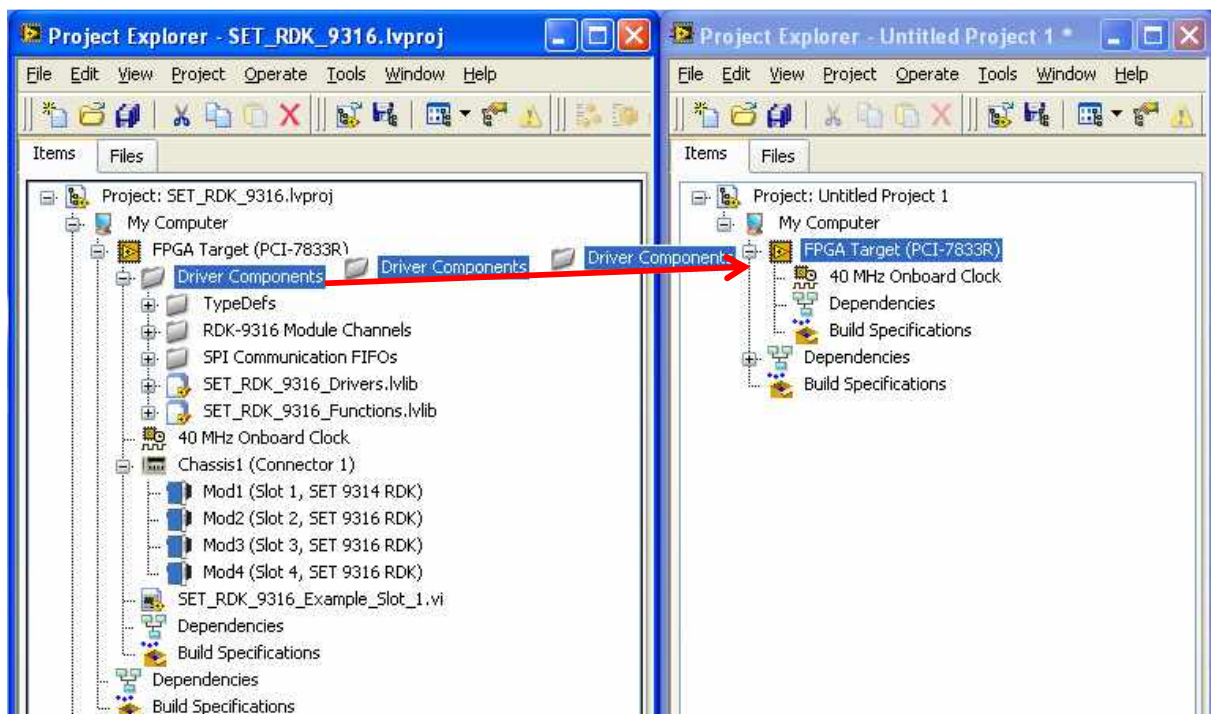


The newly added R-Series Expansion Chassis is now included in the project structure:

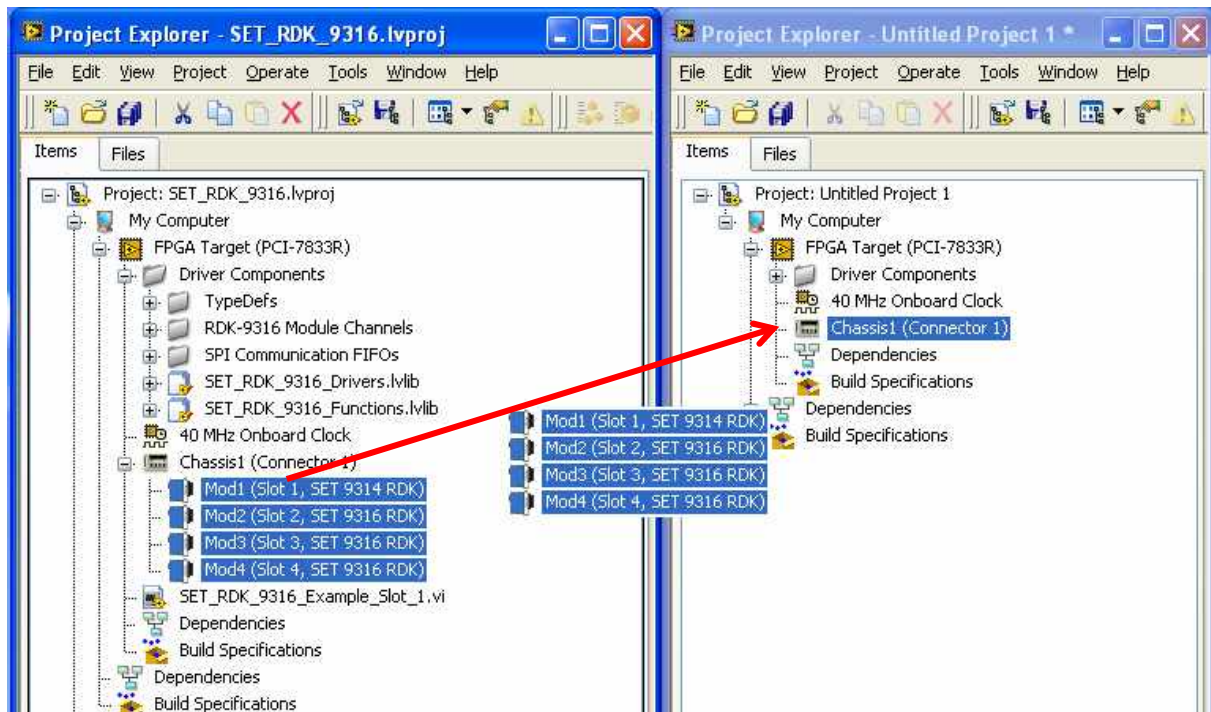


3.3.2.4 ADDING THE RDK9316 DRIVER COMPONENTS TO A PROJECT

To adopt the driver components to the new project the files must be copied **before** the RDK9316 modules are integrated.



3.3.2.5 ADDING RDK9316 MODULES TO THE PROJECT VIA DRAG-AND-DROP FUNCTION



As all driver components for all slots are already defined within the example project they will be copied with the drag-and-drop function. Components which are not used in the new application should be removed from the new project.

Caution:

Do not erase the contents of the libraries “SET_RDK_9316_Drivers.lvlib” and “SET_RDK_9316_Functions.lvlib”.

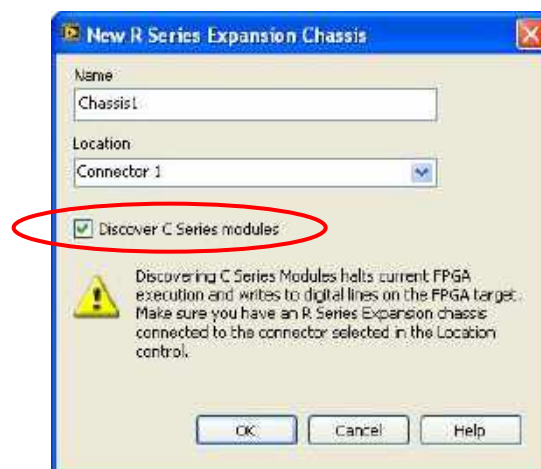
3.3.2.6 ADDING RDK9316 MODULES TO THE PROJECT VIA THE DISCOVERY-FUNCTION

The LabVIEW function “Discover C-Series Modules” automatically detects and integrates cRIO modules which are plugged into an R-Series Expansion Chassis or a cRIO-Chassis.

It is important that the I/O names of the Module match the names expected by the driver. Normally the right I/O name is created by LabVIEW within the discovery function. However on some LabVIEW versions the I/O Names are different. In this case you need to change the names of the I/O’s to the names used in the example project, or copy the I/O from the example project and delete the automatically created ones.

(PCI/PXI) FPGA-Target With R-Series Expansion Chassis:

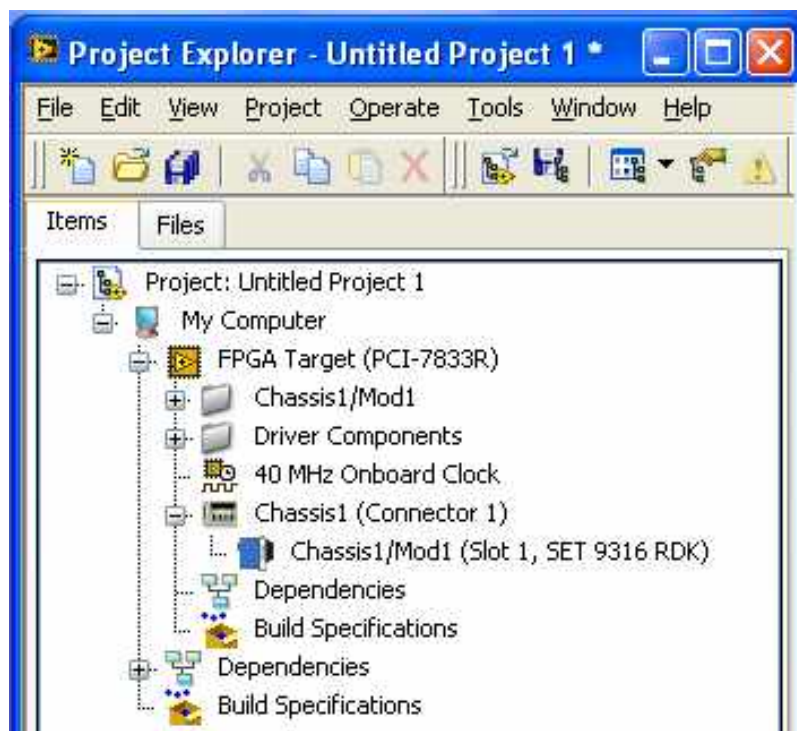
Install the new (PCI/PXI) FPGA target together with the R-Series Expansion chassis within the new project (refer to 3.3.2.2). Activate the discovery function when the following window pops up:



This initiates LabVIEW to check the chassis for cRIO modules.



When the search procedure is complete, LabVIEW visualizes the detected cRIO modules within the project structure as shown below. Additionally, for every detected module a virtual folder is automatically installed.

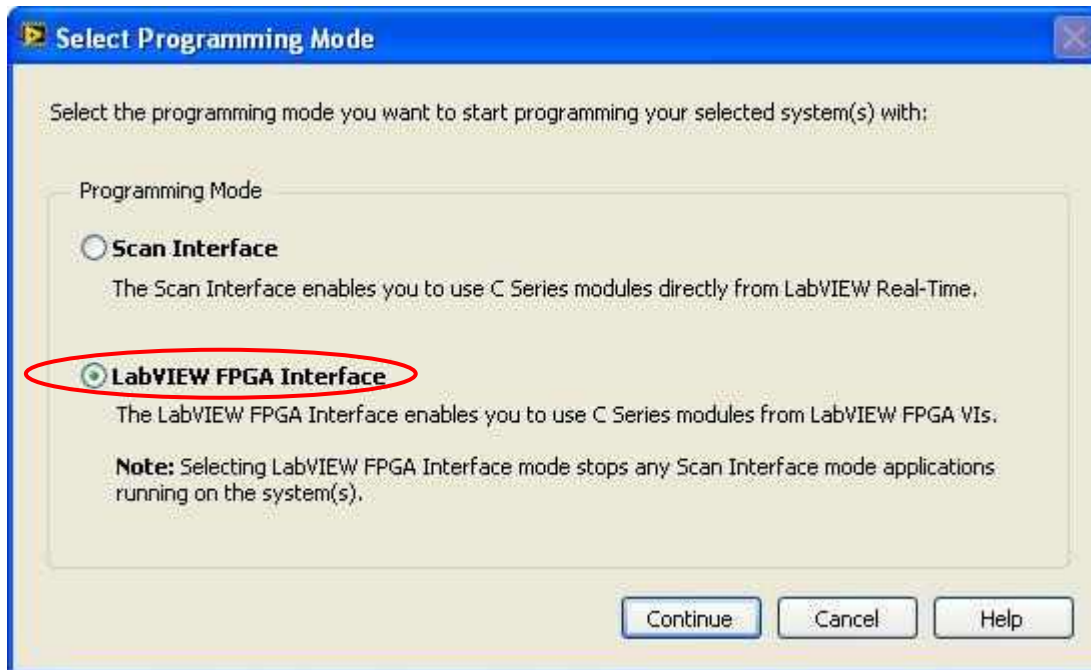


Now, the virtual folder „Driver Components“ must be copied from the example project to the new FPGA target within the LabVIEW project.

cRIO-Chassis (with FPGA-Target):

When the cRIO chassis and FPGA target does not exist in the new application, yet it must be installed as shown in chapter 3.3.2.2: select „New → Targets and Devices“ device type „cRIO Realtime“ by clicking the “+” button. LabVIEW now detects the connected cRIO chassis. On detection of the chassis it must be selected and “OK” must be checked.

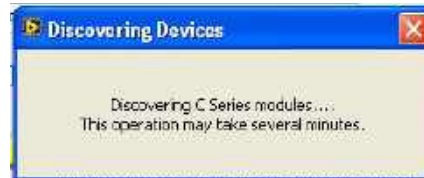
LabVIEW then asks for the I/O acquisition mode. Select method „LabVIEW FPGA Interface“.



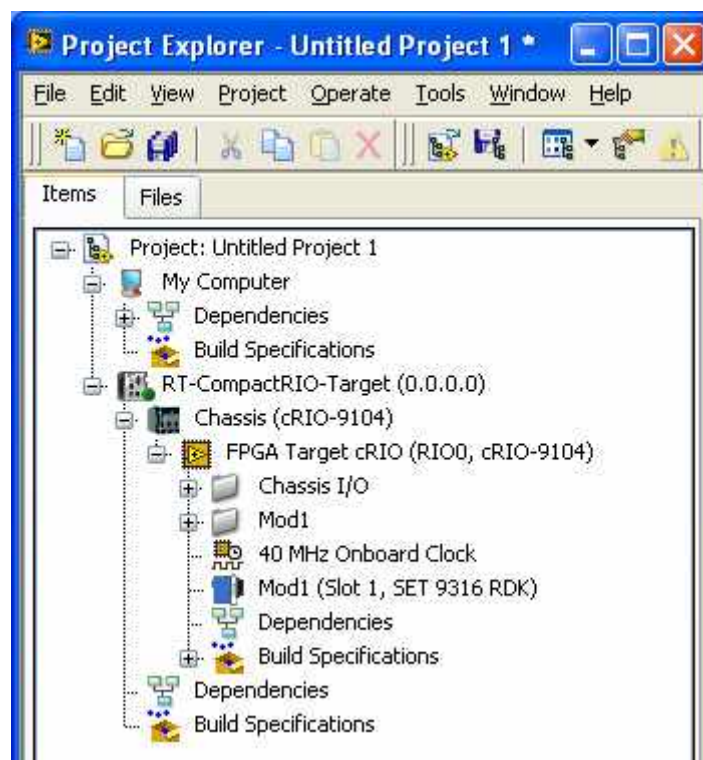
When a new chassis is added, LabVIEW automatically triggers the „Discover C-Series Modules“ procedure. Note that this procedure must be confirmed when a chassis acquisition has already taken place, previously.



LabVIEW now detects newly installed cRIO modules.
The external supply is not required for module detection.



On completion of the detection process, LabVIEW adds the detected modules to the project structure as illustrated below. Additionally, a virtual I/O-folder for every new module is automatically installed within the project.



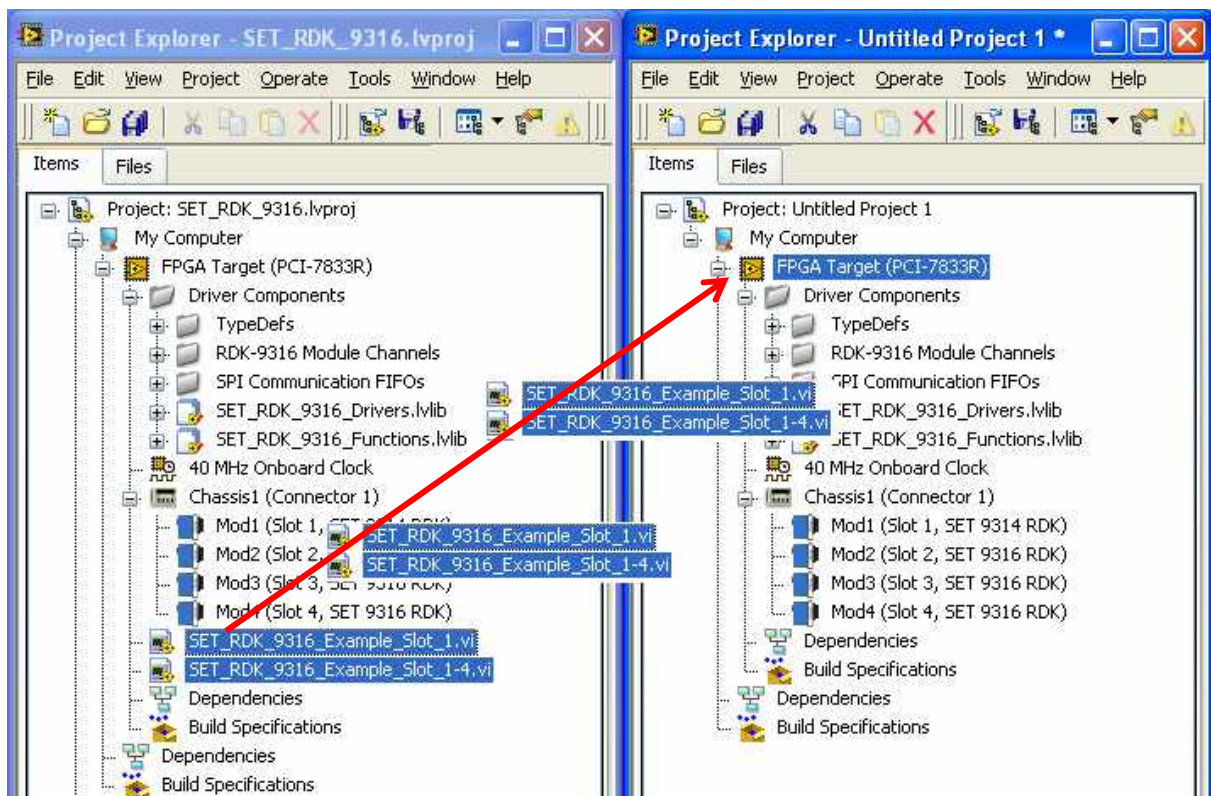
Now, the virtual folder „Driver Components“ must be copied from the example project to the new FPGA target within the LabVIEW project.

3.3.2.7 ADDING FPGA EXAMPLE-APPLICATIONS

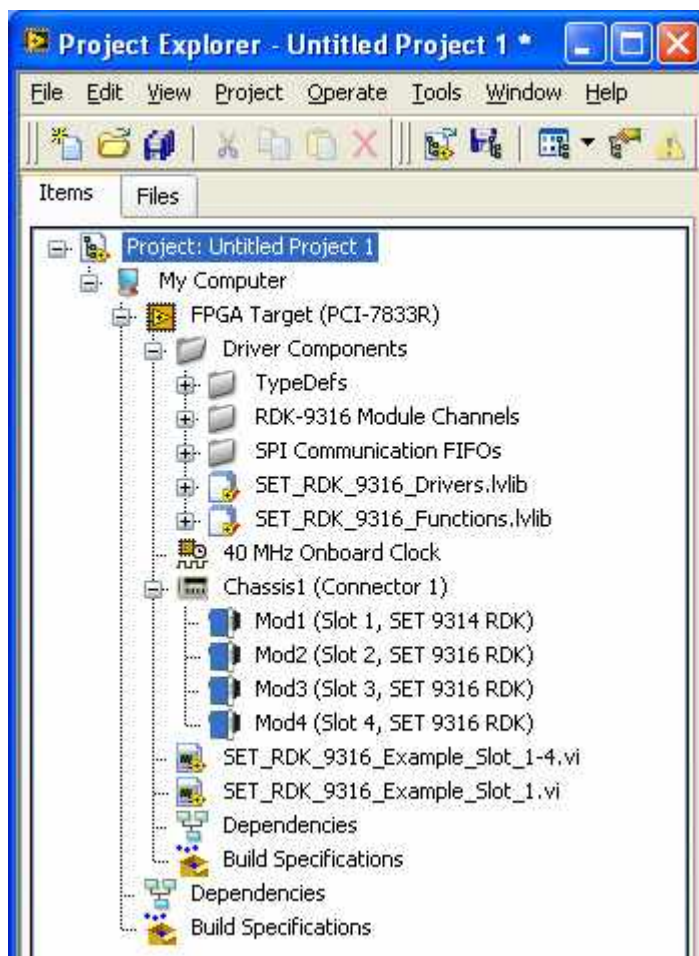
To illustrate the driver use three small example programs are shipped. The files demonstrate

- the driver application together with a single RDK9316 module installed on slot 1
- the driver application together with four RDK9316 modules
- the driver application together with eight RDK9316 modules

Use the copy & paste function to copy these examples into the new application.



All driver components and the example application are now added to the new project.

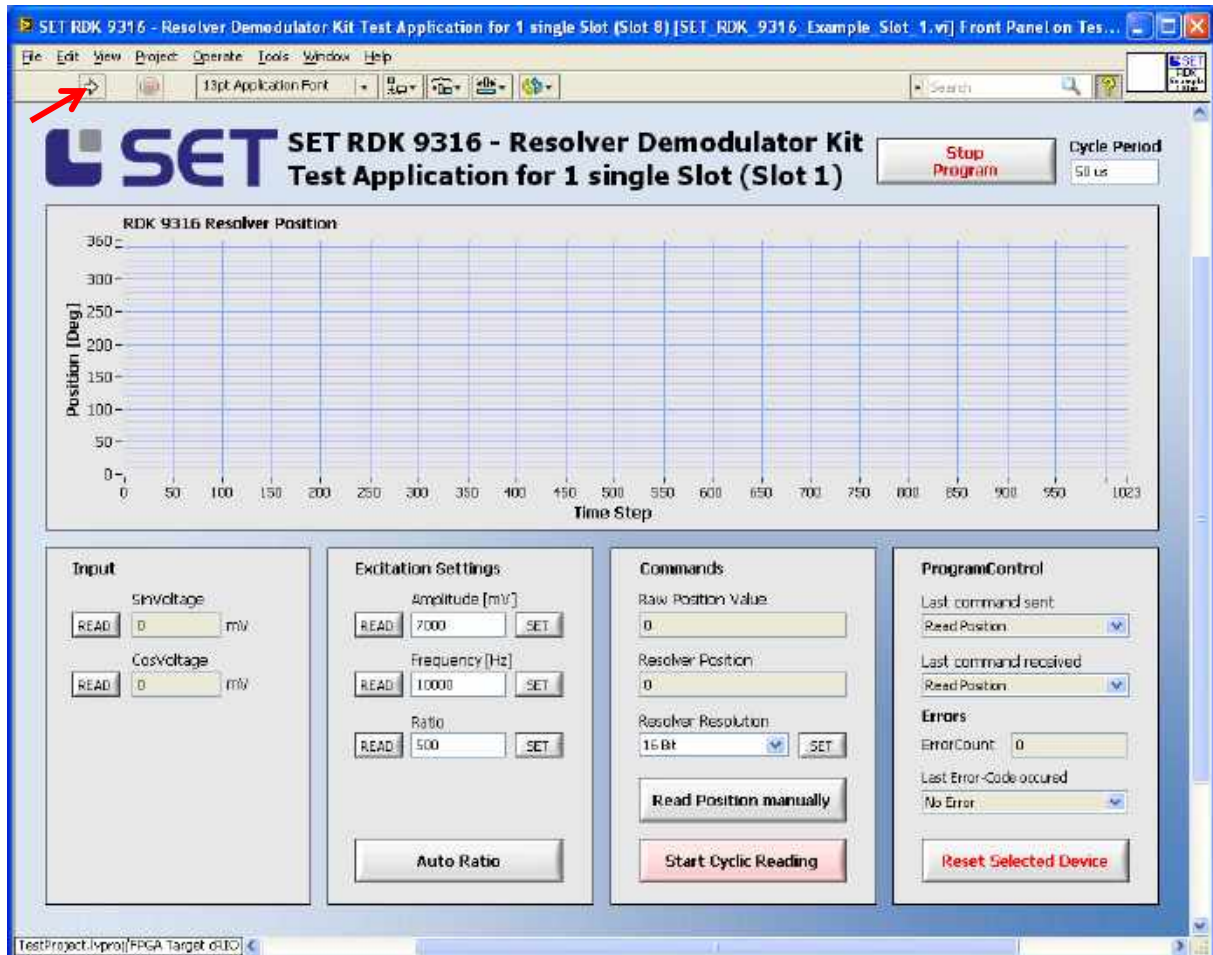


3.3.2.8 SAVING THE PROJECT

Prior to the project compilation it must be saved. To do this, „File → Save All“ must be clicked. When a new project is saved, a file name must be entered.

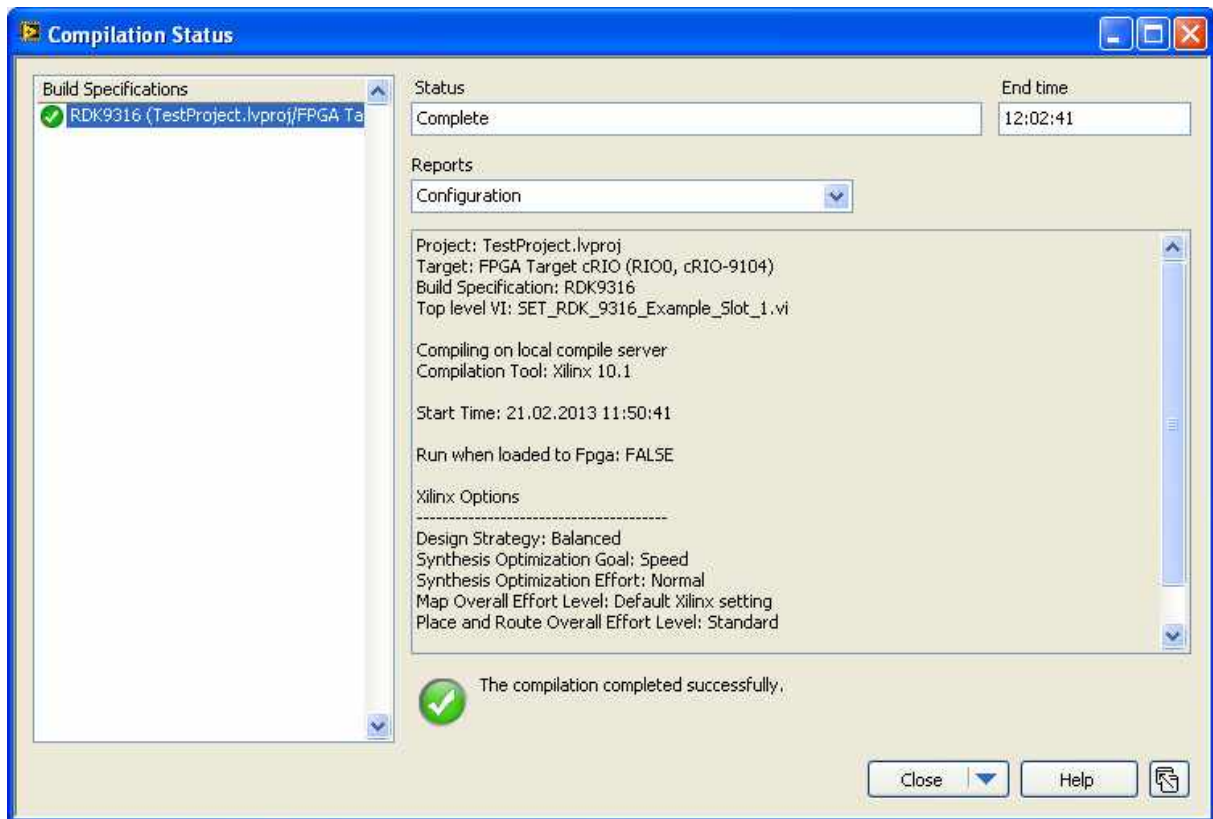
3.3.2.9 COMPILING AND RUNNING THE EXAMPLE APPLICATION

Open the „SET_RDK_9316_Example_Slot_1.vi“ file within the project structure to start the slot 1 single module application example. Then click the run button.



Now the compilation process starts and may take up to 60 minutes. Note that the number of modules used in the application has an impact on the compilation time as every module uses separate driver-VIs, function-VIs and FIFOs.

On completion of the compilation process LabVIEW visualizes the compilation results:

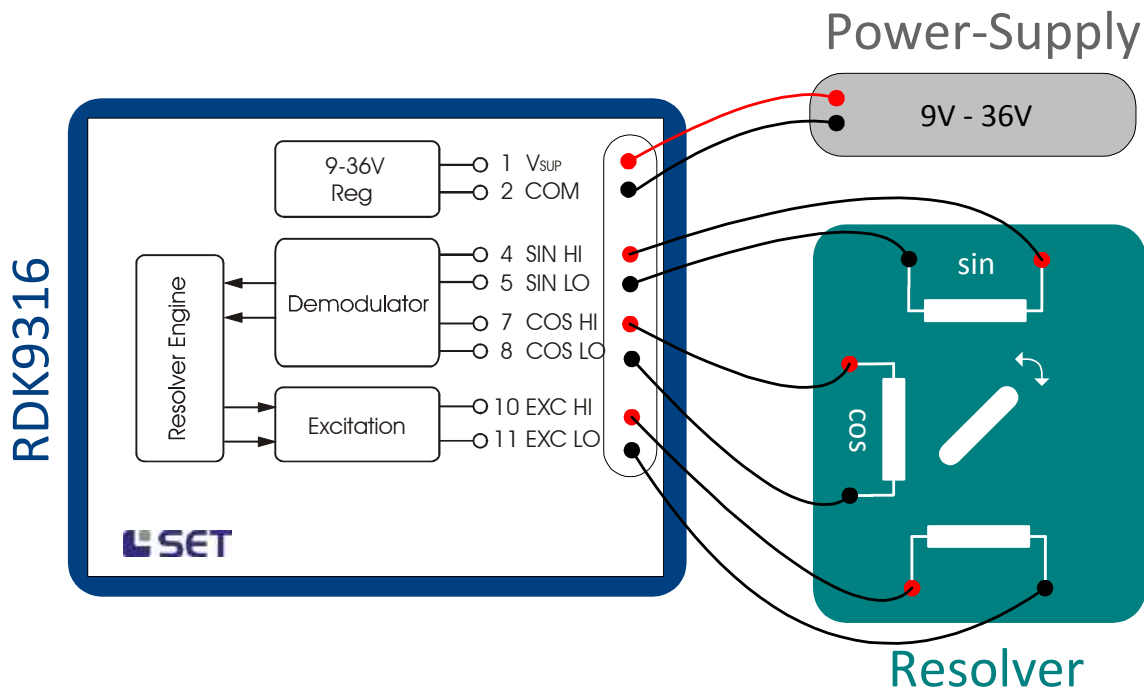


3.3.3 USING THE DRIVER TOGETHER WITH A NI REALTIME COMPACTRIO

Though the procedure illustrated before refers to a PCI-/PXI FPGA-Target, all driver components are fully compatible with a cRIO target. The only exception is the eight-slot example application which is only operable on an eight-slot cRIO chassis.

3.4 CONNECTING THE RESOLVER

To make the RDK9316 ready for use plug the module into the correct slot of the cRIO Chassis. Make sure, to select the correct slot according to the LabVIEW project definition. Then connect the modules external power supply ($9V_{DC} - 36V_{DC}$) as shown below. Connect the Resolver interface as shown below (see also paragraph 5.7 Connector Pinout).



3.5 RUNNING THE APPLICATION EXAMPLE

Make sure that:

- ✓ Application compiling process is complete
- ✓ The module is plugged into the correct slot according to the application
- ✓ The Resolver and the external supply is connected

The Application Example can now be started by clicking the “Run”-Button in LabVIEW.

Use parameters *Amplitude* [mV] and *Frequency* [Hz] to adjust the excitation signal to the correct level. For a precise demodulation process the RDK9316 must be adapted to the signal transfer ratio of the Resolver. This is accomplished by executing the function “*Auto Ratio*” which automatically adjusts the RDK9316 module to the Resolver transfer ratio. Note that the Resolver must be connected to the module for this process. When the Resolver type is changed, make sure to execute “*Auto Ratio*” again.

4. APPLICATION DEVELOPMENT

4.1.1 RDK9316 DRIVER AND COMMUNICATION

The RDK driver package includes eight driver-VIs, eight functions-VIs and 16 communication-FIFOs which are used internally. The driver- and function-VIs are located in two different libraries, „SET_RDK_9316_Drivers.lvlib“ and „SET_RDK_9316_Functions.lvlib“. All driver components are located in the project structure of the shipped example projects (folder “Driver Components”).

RDK9316 Driver-VI’s:

The driver VI’s control the SPI communication between the FPGA and the RDK9316 modules and operate as server. These VI’s wait for an application call. On reception of an instruction they communicate with the modules via SPI and return the response data to the calling process.

RDK9316 Functions-VI’s:

The Functions-VI’s provide the user interface for the application and transfer instructions and data to the driver VI’s. These VI’s act as clients for the transfer instructions and data to the driver-VI’s.

4.1.1.1 DRIVER-VI IMPLEMENTATION

The implementation of the RDK9316 drivers into a LabVIEW application takes place by simply locating the relevant VI’s on the main programs block diagram. Each chassis slot uses its own driver-VI.

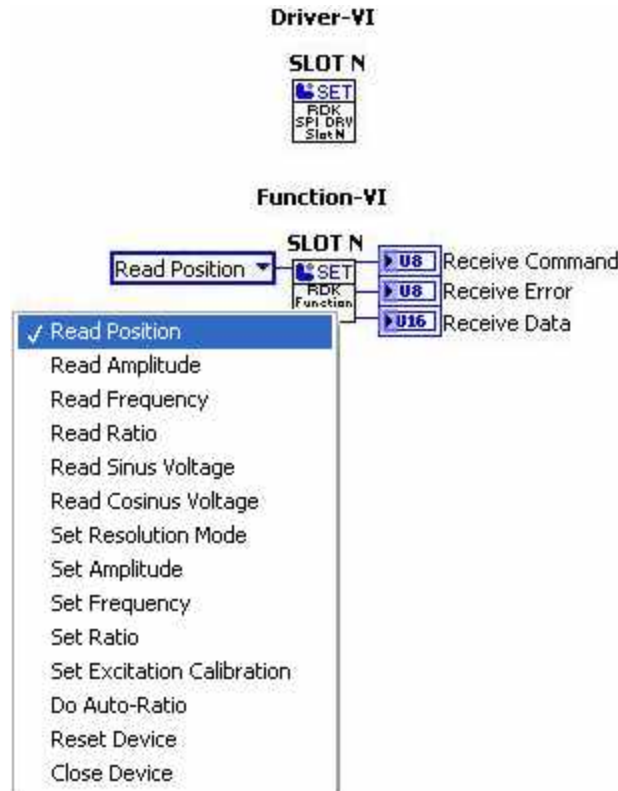


When the LabVIEW application is started, all involved driver-VI’s are also processed. The driver-VI’s act as server and wait for instructions from the application.

Important notice: As the driver VI’s are processed continuously when the application program is operated, it is important to locate them in such a way as not to jam up the rest of the program code. When the driver-VI’s are used inside a loop or a sequence please note that all included driver-VI’s must be closed before the loop or sequence continues.

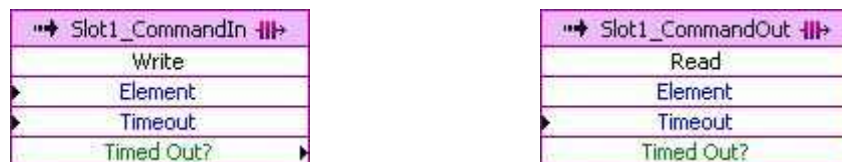
4.1.1.2 TRIGGERING RDK9316 FUNCTIONS VIA FUNCTIONS-VI

The functions-VI's are located in the „SET_RDK_9316_Functions.lvlib“. Note that every chassis slot requires a specific functions-VI.



Calling a RDK9316 function via the Functions-VI

When a Function-VI is executed, instructions and data are transferred to the applicable FIFOs „SlotN_CommandIN“. The applicable driver-VI then communicates with the RDK9316 module via the driver internal FIFO „SlotN_CommandOUT“.



Driver internal FIFOs for data transfer between Function-VI and Driver-VI (Slot1 example)

4.1.1.3 INITIALISING AND CLEARING THE FIFO BUFFERS

To ensure a correct data transfer it is essential to clear the FIFO contents during the programs initialisation sequence. This is accomplished by executing the “Clear” function after program start.

Important notice: Execute the “Clear” instruction before the drivers (which are part of the block diagram) are operated.

The example below illustrates the FIFO buffer clearing of an application which uses eight modules:

Clear all used Communication-FIFOs on Startup

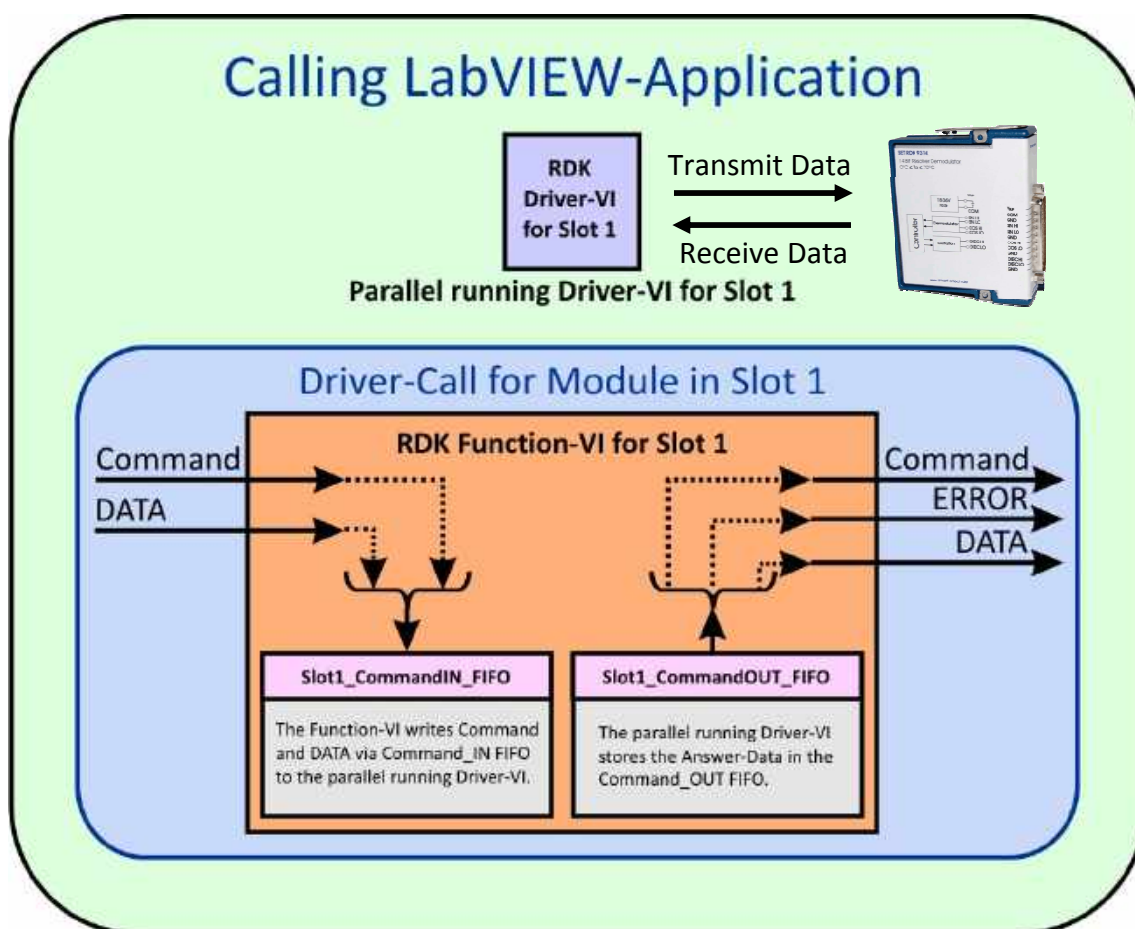
Slot1_CommandIn Clear	Slot1_CommandOut Clear
Slot2_CommandIn Clear	Slot2_CommandOut Clear
Slot3_CommandIn Clear	Slot3_CommandOut Clear
Slot4_CommandIn Clear	Slot4_CommandOut Clear
Slot5_CommandIn Clear	Slot5_CommandOut Clear
Slot6_CommandIn Clear	Slot6_CommandOut Clear
Slot7_CommandIn Clear	Slot7_CommandOut Clear
Slot8_CommandIn Clear	Slot8_CommandOut Clear

Clearing all (used) communication FIFOs during program start

4.1.1.4 APPLICATION TO MODULE COMMUNICATION DIAGRAM

Communication with a single RDK9316 module:

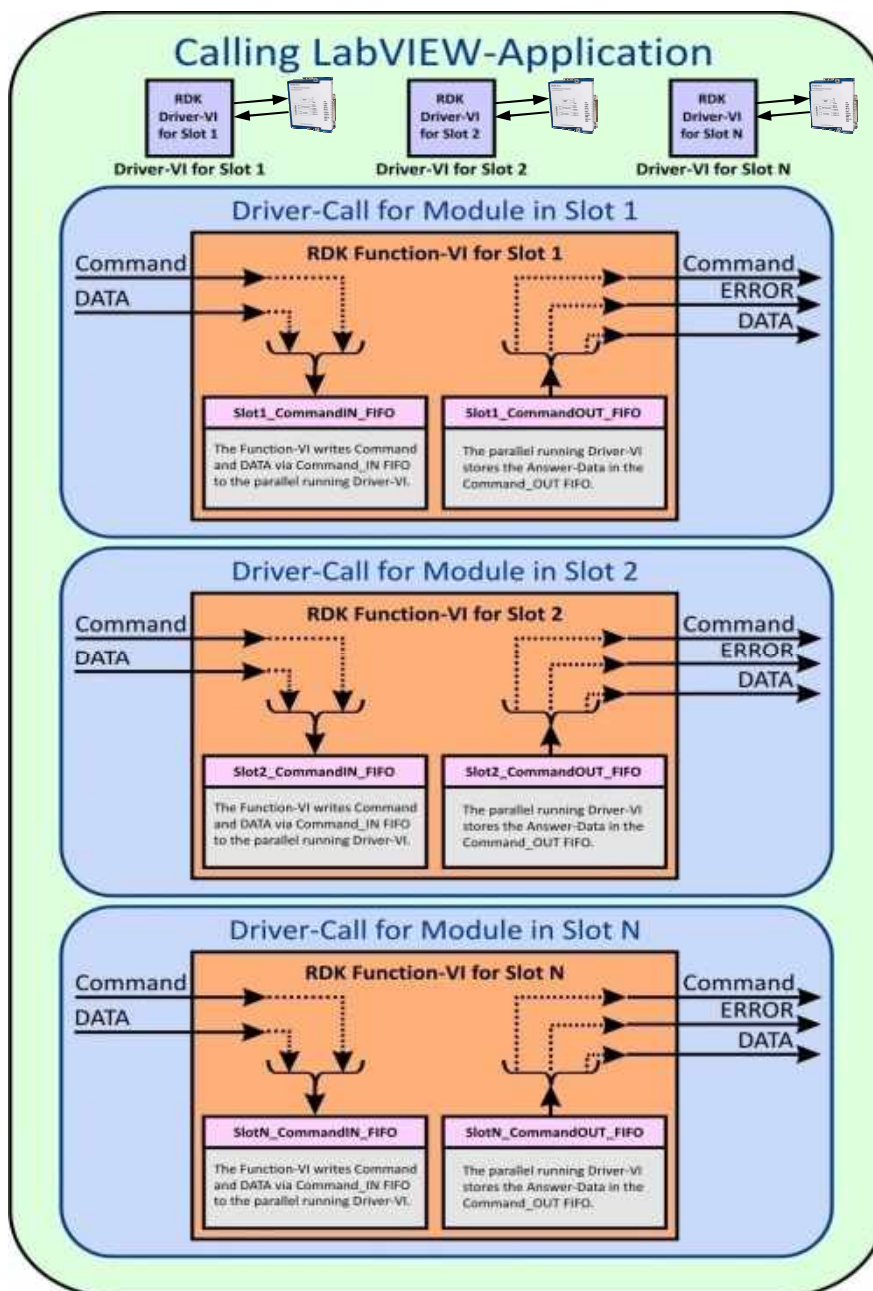
The diagram below illustrates the communication scheme between application, Function-VI, Driver-VI and RDK9316 module (slot 1). From the diagram it can be noticed that the Driver-VI of the FPGA driver (slot 1) is located outside and parallel to the code section which triggers the instructions. However, the Function-VI may be used within any LabVIEW structure like loops, cases or sequences. Furthermore, multiple calls are allowed which then are operated sequentially. Note that the Function-VIs use non-re-entrant structures.



Communication between application and RDK9316 module

Communication with multiple RDK9316 modules:

The diagram below illustrates the communication scheme between application, Function-VI, Driver-VI and multiple RDK9316 modules. From the diagram it can be noticed that the Driver-VIs of the FPGA drivers are located outside and parallel to the code section which triggers the instructions. However, the Function-VI may be used within any LabVIEW structure like loops, cases or sequences. Furthermore, multiple calls are allowed which then are operated sequentially. Note that the Function-VIs use non-re-entrant structures.



Communication between application and multiple RDK9316 modules

4.2 RDK9316 DRIVER-INSTRUCTIONS

The RDK9316 driver instructions are defined within the type definition file „USRCMD.ctf“. Refer to this file to note the ring type structure. The instruction names thus correlate with an identification number which is given in brackets (), below.

Command execution requires a specific execution time and varies due to asynchronous processing principles. Hence a timeout must be defined for an instruction call which by default is 40.000.000 system ticks (equal to 1 second for a 40MHz FPGA). When no timeout is connected to the input terminal of the Function-VI, the default timeout value is active. To specify a timeout longer than the default value, the input must be connected to a new timeout value. Note that specifying a timeout less than 40.000.000 has no effect because the default value cannot be reduced. If an instruction processing takes longer than the specified timeout, a fault is generated from the driver (refer to the drivers fault codes).

<u>Instruction [U8]:</u>	<u>Tx-Data [U8]:</u>	<u>Receive-Data [U16]:</u>	<u>Data Sink:</u>
Read Position (1)	no data	Resolver Position [RAW]	RDK9316 Controller



This instruction reads the actual resolver position. The data scaling is binary data within the range of 0..65535 equal to 0°..360-(360/65536)°. Note that the 0..65535 data range is independent from the selected resolver resolution. It takes 24us (+/-1us) to receive the position data; i.e. the resolver position has already changed when the position data is received. This “position delay” can be calculated by

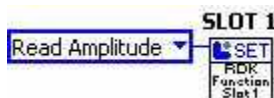
$$\Delta \text{Position [arcmin]} = \text{Angular Speed [arcmin/s]} * 24\mu\text{s}$$

The maximum position sampling rate for cyclic reading is 40kHz. To calculate the resolver position in engineering units use the following equation:

$$\varphi_{DEG} = \text{DATA} \cdot \frac{360}{65536}$$

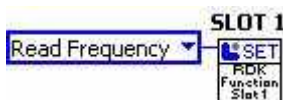
Note that dividing by 65536 equals a logical shift of 16!

<u>Instruction [U8]:</u>	<u>Tx-Data [U8]:</u>	<u>Receive-Data [U16]:</u>	<u>Data Sink:</u>
Read Amplitude (3)	no data	Excit. Amplitude. [mV _{RMS}]	RDK9316 Controller



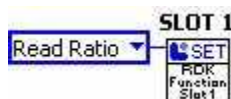
This instruction reads the programmed amplitude of the excitation voltage. The return data format is a 16bit engineering value in [mV_{RMS}].

<u>Instruction [U8]:</u>	<u>Tx-Data [U8]:</u>	<u>Receive-Data [U16]:</u>	<u>Data Sink:</u>
Read Frequency (4)	no data	Excit. Frequency. [Hz]	RDK9316 Controller



This instruction reads the programmed frequency of the excitation voltage. The return data format is a 16bit engineering value in [Hz].

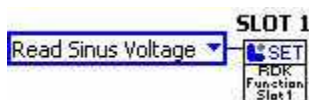
<u>Instruction [U8]:</u>	<u>Tx-Data [U8]:</u>	<u>Receive-Data [U16]:</u>	<u>Data Sink:</u>
Read Ratio (5)	no data	Resolver Ratio [¹ / ₀₀]	RDK9316 Controller



This instruction reads the programmed resolver transformation ratio (RAT). The return data format is a 16bit engineering value with dimension [¹/₀₀].

$$\begin{aligned} \text{RAT} = 100 \text{ [}^1\text{/}_{00}\text{]} & \text{ equal transformation ratio} = 0.1 \\ \text{RAT} = 1000 \text{ [}^1\text{/}_{00}\text{]} & \text{ equal transformation ratio} = 1.0 \end{aligned}$$

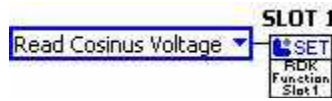
<u>Instruction [U8]:</u>	<u>Tx-Data [U8]:</u>	<u>Receive-Data [U16]:</u>	<u>Data Sink:</u>
Read Sine Input (6)	no data	Sine Input Voltage [mV _{RMS}]	RDK9316 Controller



This instruction reads the sine input channel voltage amplitude filtered with 50Hz. The return data format is a 16bit engineering value with dimension [mV_{RMS}]. This value is NOT for Position calculation and updated only 9 times per Second

<u>Instruction [U8]:</u>	<u>Tx-Data [U8]:</u>	<u>Receive-Data [U16]:</u>	<u>Data Sink:</u>
--------------------------	----------------------	----------------------------	-------------------

Read Cos Input (7) no data Cos Input Voltage [mV_{RMS}] RDK9316 Controller



This instruction reads the sine input channel voltage amplitude filtered with 50Hz. The return data format is a 16bit engineering value with dimension [mV_{RMS}]. This value is NOT for Position calculation and updated only 9 times per Second

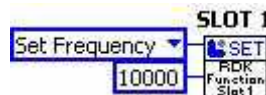
Instruction [U8]: Tx-Data [U8]: Receive-Data [U16]: Data Sink:
Set Amplitude (9) Excit. Amp.[mV_{RMS}] Excit. Amp.[mV_{RMS}] RDK9316 Controller



This instruction programs the excitation amplitude. The data format is a 16bit engineering value with dimension [mV_{RMS}].

Valid Amplitude Range: 2000 [mV_{RMS}].. 7000 [mV_{RMS}]

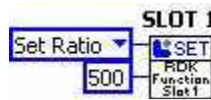
Instruction [U8]: Tx-Data [U8]: Receive-Data [U16]: Data Sink:
Set Frequency (10) Excit. Frequ.[Hz] Excit. Frequ.[Hz] RDK9316 Controller



This instruction programs the excitation frequency. The data format is a 16bit engineering value with dimension [Hz].

Valid Frequency Range: 1000 [Hz].. 10000 [Hz] for 14 and 16Bit
 2000 [Hz].. 10000 [Hz] for 10 and 12Bit

<u>Instruction [U8]:</u>	<u>Tx-Data [U8]:</u>	<u>Receive-Data [U16]:</u>	<u>Data Sink:</u>
Set Ratio (11)	Resolver Ratio[⁰ / ₀₀]	Resolver Ratio[⁰ / ₀₀]	RDK9316 Controller



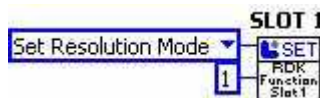
This instruction programs the resolver ratio. The data format is a 16bit engineering value with dimension [⁰/₀₀].

Valid Ratio Range: 100 [⁰/₀₀].. 1000 [⁰/₀₀]

Example: 100 [⁰/₀₀] equal transformation ratio = 0,1
1000 [⁰/₀₀] equal to transformation ratio = 1,0

Important notice: It is important to program the correct resolver ratio to get the full RDK9316 precision.

<u>Instruction [U8]:</u>	<u>Tx-Data [U8]:</u>	<u>Receive-Data [U16]:</u>	<u>Data Sink:</u>
Set Resolution Mode (8)	Res.Mode	Res.Mode	RDK9316 Controller



This instruction programs the position resolution. The data format is enumeration according to the following table.

Valid position resolution data:

- Data = 0, equal to High Resolution (16 bit)
- Data = 1, equal to Normal Resolution (14 bit)
- Data = 2, equal to High Speed (12 bit)
- Data = 3, equal to High Speed (10bit)

Note: The command fails if changing to 12 or 10 Bit while having a Excitation Frequency below 2000Hz. Change the frequency first to a value >= 2000Hz if setting Resolution to 12 or 10 Bit

<u>Instruction [U8]:</u>	<u>Tx-Data [U8]:</u>	<u>Receive-Data [U16]:</u>	<u>Data Sink:</u>
Do Auto Ratio (14)	No Data	Adjusted Ratio	RDK9316 Controller



This instruction initiates the automatic resolver adaption (refer to chapter4.6.) and returns the evaluated resolver ratio.

Valid Ratio Range: 100 [⁰/₀₀].. 1000 [⁰/₀₀]

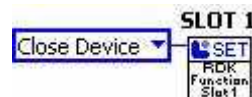
Example: 100 [°/00] equal transformation ratio = 0,1
 1000 [°/00] equal to transformation ratio = 1,0

<u>Instruction [U8]:</u>	<u>Tx-Data [U8]:</u>	<u>Receive-Data [U16]:</u>	<u>Data Sink:</u>
Reset Device (15)	No Data	No Data	RDK9316 Controller



This instruction resets the RDK9316 module. Note that a module reset requires appr. 100ms processing time. When data is sent to the module while the module is processing a reset the module fault “NO MODULE” will occur.

<u>Instruction [U8]:</u>	<u>Tx-Data [U8]:</u>	<u>Receive-Data [U16]:</u>	<u>Data Sink:</u>
Close Device (16)	No Data	No Data	FPGA Driver-VI




This instruction stops and terminates the applicable FPGA driver-VI.

Note: when the application is terminated or the actual part of code (which includes the driver-VI's) is left, each driver-VI which was previously launched must be terminated with the “Close Device” instruction before.

4.3 RDK9316 DRIVER ERROR CODES

Error Code [U8]	Error Type	Error Detection	Description
0	No Error	none	If no error has appeared during communication with the RDK9316 module, the received telegram has error code „0“.
1	Unknown Command	RDK9316 module	This error occurs when the RDK9316 module receives an unknown instruction code.
2	Communication Error	RDK9316 module	This error occurs when the FPGA aborts a telegram transmission to the RDK9316 module or when the SPI timing constraints are exceeded.
3	Invalid Parameter	RDK9316 module	This error occurs when the parameter “Data” is out of range.

Error Code [U8]	Error Type	Error Detection	Description
8	Excitation ShortCircuit	RDK9316 module	This error occurs if the Excitation is in Short Circuit and deactivated due to thermal limits. The Excitation will be disabled for 5 seconds and the Error will remain 10 more seconds. Regularly driving the Excitation in its thermal limit will degrade performance!
100	No Module	FPGA Driver-VI	This error occurs when no RDK9316 module is connected to the applicable slot. Note that this error also occurs when module access takes place while the module processes a reset instruction.
101	Module Busy	Functions-VI	The busy signal to the FPGA indicates module readiness. When a time consuming instruction is processed by the module, the busy signal becomes valid for a certain amount of time. However, if a new instruction is sent to the module while the module still processes a previous instruction (and therefore the busy line still is active), the FPGA driver waits for the busy line to return to the ready state. The maximum idle time for the FPGA driver is a default value and can be increased from the user. When the specified timeout is exceeded, the error "Module Busy" is generated by the functions-VI.
103	Transmit Incomplete	FPGA Driver-VI	This error occurs when the RDK9316 module aborts telegrams or exceeds the SPI timing constraints.
104	Wrong SPI Answer	Functions-VI	This error occurs when the replied telegram does not match with the instruction telegram. Note: This error may occur subsequently to a module power interrupt while the application is still running. However, this is only a singular error and only possible during the execution of a position read instruction.
105	FIFO Timeout	Functions-VI	This error occurs if a FIFO fault takes place during data transfer from the functions-VI to the applicable driver-VI. Note: To reset this error scenario, both communication FIFOs (SlotN_CommandIN und SlotNCommandOUT) of the applicable module must be cleared with the „Clear“ instruction as shown below: 

Error Code [U8]	Error Type	Error Detection	Description
106	Power Error	Functions VI	This Error occurs if the external Supply of the Module is not in it's limits.
107	Loose of Tracking	RDK9316 Module	This error occurs if one of the following conditions exists: <ul style="list-style-type: none"> - Loose of Signal : Sin and Cos Inputs both less than 500mV - Loose of Tracking: Resolver is moving to fast, the error is over ~100 LSB - 180° phase error detection (false null). (corrected after detection)

4.4 MODULE IDENTIFICATION UNDER LABVIEW

LabVIEW automatically detects the cRIO module type (Discover C-Series Modules), even if the external Supply is not applied. Please see Chapter 3.3.2.6 for further instructions.

4.5 SAVING THE SETUP

The RDK9316 automatically saves the setup in a non-volatile memory. After a power interruption the last set Parameters are active when the Module is switched on.

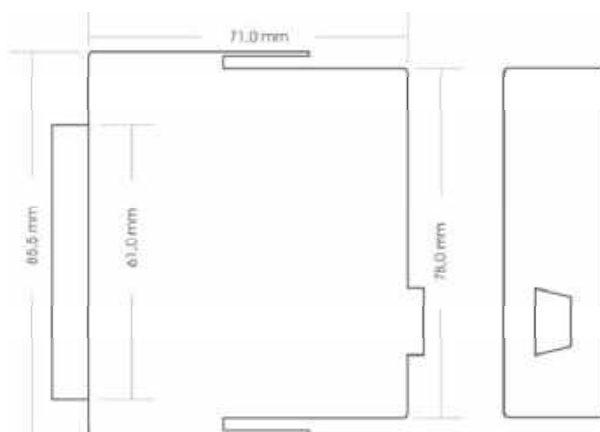
4.6 AUTO RATIO

The Auto Ratio function (Do Auto-Ratio) automatically adjusts the RDK9316 to the transfer ratio of the connected Resolver. The Auto Ratio must be triggered when the type of Resolver is changed or a Resolver is connected to the module for the first time. Note that position readings may be incorrect or out of tolerance when the module is not adjusted to the Resolver transfer ratio.

5. TECHNICAL SPECIFICATION

5.1 HOUSING

- NI CompactRIO 9952 Standard Housing
- Dimensions: appr. 71mm x 72mm x 23mm



5.2 EXTERNAL POWER SUPPLY

- Supply Voltage: $9V_{DC} - 36V_{DC}$
- Power Consumption: max. 1.5 Watt without Resolver

Caution:

The external power supply input is isolated from the cRIO chassis signals with a maximum isolation voltage of $50V_{DC}$. Note that the excitation output and the Sine/Cosine input signals are galvanically connected to the external power supply.

5.3 EXCITATION OUTPUT

- Frequency: $1\text{kHz} - 10\text{kHz}$, adjustable via software (14 / 16Bit)
 $2\text{kHz} - 10\text{kHz}$, adjustable via software (10 / 12Bit)
- Amplitude: $2V_{RMS} - 7V_{RMS}$, adjustable via software
- Current: max. 120mA_{RMS}

Caution:

The excitation output can be damaged when a power source is connected to its terminals. Driving the excitation regularly in short circuit condition will degrade performance!

5.4 SINUS AND COSINE SIGNAL INPUTS

- Voltage Range: $2V_{RMS} - 7V_{RMS}$
- Zin (differential): 54kOhm
- Zin (single-ended): 27kOhm

Caution:

The signal inputs can be damaged, when the input voltage exceeds a limit of $40V_{peak}$.

5.5 POSITION PROCESSING

- Resolution: 10/12/14/16Bit, adjustable via software
- Accuracy: up to 5.6 Arc min
- Settling Time (179°-step): 2 ms (10Bit) 8ms (12Bit)
20ms (14Bit) 50ms(16Bit)
- Tracking Rate: 1 600 rpm (16Bit) 3 700 rpm(14Bit)
15 000 rpm (12Bit) 30 000 rpm(10Bit)

5.6 ENVIRONMENTAL CONDITIONS

- Temperature Range: $0^{\circ}C - 70^{\circ}C$
- Humidity: 10% – 90% relative, non-condensing

5.7 CONNECTOR PINOUT

User Interface Connector (25-pin Sub-D / male):

Pin	Function
1	V _{SUP}
2	COM
3	GND
4	+SIN
5	-SIN
6	GND
7	+COS
8	-COS
9	GND
10	+EXC
11	-EXC
12 - 25	n.c.

6. MODULE CALIBRATION

The RDK9316 is calibrated when shipped. A recalibration can be done by SET GmbH.

7. MODULE MAINTENANCE

No maintenance is required for the RDK9316 module.

8. SERVICE ADDRESS

For technical support contact the SET service address:

SET GmbH
August-Braun-Str. 1
88239 Wangen/Allgäu
Germany
Tel.: +49 (0)7522 91687-600
Fax: +49 (0)7522 91687-899
e-Mail: support.crio@smart-e-tech.de