REXYGEN www.rexygen.com

Database driver for ODBC in the REXYGEN system (`DbDrv` driver) User guide

REX Controls s.r.o.

Version 3.0.3
Plzeň (Pilsen), Czech Republic
2024-11-03

Contents

Chapter 1

The DbDrv Driver and the REXYGEN System

1.1 Introduction

This manual describes the `DbDrv` driver for connecting the REXYGEN system to various databases via an ODBC interface. This driver facilitates both reading from and writing to the database. Additionally, it enables the export of the REXYGEN system's archives, including alarms, events and trends.

1.2 System Requirements

The `DbDrv` driver is compatible with both Windows and Linux target devices. A TCP/IP stack (Ethernet card, USB WiFi dongle, etc.) is required for network communication. An appropriate ODBC driver is required for the specific database system (e.g., MySQL, PostgreSQL, MS-SQL, etc.).

For effective use of the driver, the following software must be installed on both the host (development) and target (runtime) computers:

Development Computer	
Operating System	One of the following: Windows 10/11, GNU/Linux
Development Tools	Corresponding version of the REXYGEN system development tools
Target Device	
REXYGEN System	Runtime core for the chosen operating system
IO Driver	Appropriate version for the operating system
ODBC Interface	Specific version for the chosen database (MySQL, PostgreSQL, MS-SQL, etc.)

1.3 Installation of the Driver on the Host Computer

The DbDrv driver is included in the REXYGEN system's Development tools installation package. During installation, ensure to select the relevant package. The REXYGEN system typically installs in the following directory: `C:\Program Files\REX Controls\REXYGEN <version>`.

The installation process copies the following files to the installation folder:

`Bin\DbDrv_H.dll` – The configuration component of the DbDrv driver.

`Bin\DbDrv_T.dll` – The target component of the DbDrv driver, invoked by the RexCore runtime module.

`Doc\PDF\ENGLISH\DbDrv_ENG.pdf` – This user manual.

1.4 Installation of the Driver on the Target Device

1.4.1 Windows Machines

The target component of the driver for connecting to the database on Windows 10/11 is included in the REXYGEN system's Development tools, as previously mentioned.

1.4.2 Linux Machines

If the RexCore runtime module is not installed on your target device, refer to the Getting Started guide of the REXYGEN system [1] for installation instructions. This includes all necessary drivers, including DbDrv.

To install DbDrv separately on Linux, use the following command:

```
sudo apt-get install rex-dbdrvt rex-odbc
```

1.5 Installation of the ODBC Interface on the Target Device

For ODBC connections, it is essential to install and configure the ODBC interface for the corresponding database system on the target device, regardless of the operating system.

On Debian Linux, the `rex-odbc` package automatically installs and configures the ODBC interface for:

- MySQL (package `libmaodbc` or `odbc-mariadb`), ODBC driver name `MySQL`,
- MariaDB (package `libmaodbc` or `odbc-mariadb`), ODBC driver name `MariaDB`,
- Microsoft SQL (MSSQL) (package `tdsodbc`), ODBC driver name `MSSQL`,
- PostgreSQL (package `odbc-postgresql`), ODBC driver name `PostgreSQL`.

No additional manual configuration is typically required. The following sections provide specific installation details for each database system.

1.5.1 Debian Linux – MySQL

The necessary packages are `unixodbc` and `libmaodbc` or `odbc-mariadb`. Install them using:
`sudo apt-get install unixodbc odbc-libmaodbc`

Further it is necessary to append the following section to the `/etc/odbcinst.ini` file:

```
[MySQL]
Description     = MySQL driver
Driver          = libmaodbc.so
```

1.5.2 Debian Linux – MariaDB

The necessary packages are `unixodbc` and `libmaodbc` or `odbc-mariadb`. Install them using:
`sudo apt-get install unixodbc odbc-mariadb`

Further it is necessary to append the following section to the `/etc/odbcinst.ini` file:

```
[MariaDB]
Description     = MariaDB driver
Driver          = libmaodbc.so
```

1.5.3 Debian Linux – PostgreSQL

The necessary packages are `unixodbc` and `odbc-postgresql`. Install them using:
`sudo apt-get install unixodbc odbc-postgresql`

Further it is necessary to append the following section to the `/etc/odbcinst.ini` file:

```
[PostgreSQL ANSI]
Description       = PostgreSQL ODBC driver (ANSI version)
Driver            = psqlodbca.so
Setup             = libodbcpsqlS.so

[PostgreSQL Unicode]
Description       = PostgreSQL ODBC driver (Unicode version)
Driver            = psqlodbcw.so
Setup             = libodbcpsqlS.so
```

1.5.4 Debian Linux – Microsoft SQL (MSSQL)

The necessary packages are `unixodbc` and `tdsodbc`. Install them using:
`sudo apt-get install unixodbc tdsodbc`

Further it is necessary to append the following section to the `/etc/odbcinst.ini` file:

```
[MSSQL]
Description     = Microsoft SQL (FreeTDS) Driver
Driver          = libtdsodbc.so
Setup           = libtdsS.so
```

It is recommended to always define PORT value in connection string with Microsoft SQL driver, because the default value can vary with ODBC adapter build configuration.

1.5.5 Debian Linux – Common ODBC DSN Configuration

Optionally, connection parameters can be stored under a specified name – a *DSN* – in the `/etc/odbc.ini` file:

```
[MyDSN]
Driver       = MSSQL
Description  = Microsoft SQL server - My great application
SERVER       = sqlsrv.example.com
PORT         = 1433
Database     = MyDatabase
```

Then the *connection-string* have to be specified in form:
`DSN=MyDSN;UID=username;PWD=password;`.

Putting username and password into DSN configuration in `/etc/odbc.ini` is usually not supported (depends on database system driver).

1.5.6 Other platforms and database systems

Instructions on installing the ODBC driver on your platform should be included in the documentation of your database system (MySQL, PostgreSQL, Microsoft SQL etc.).

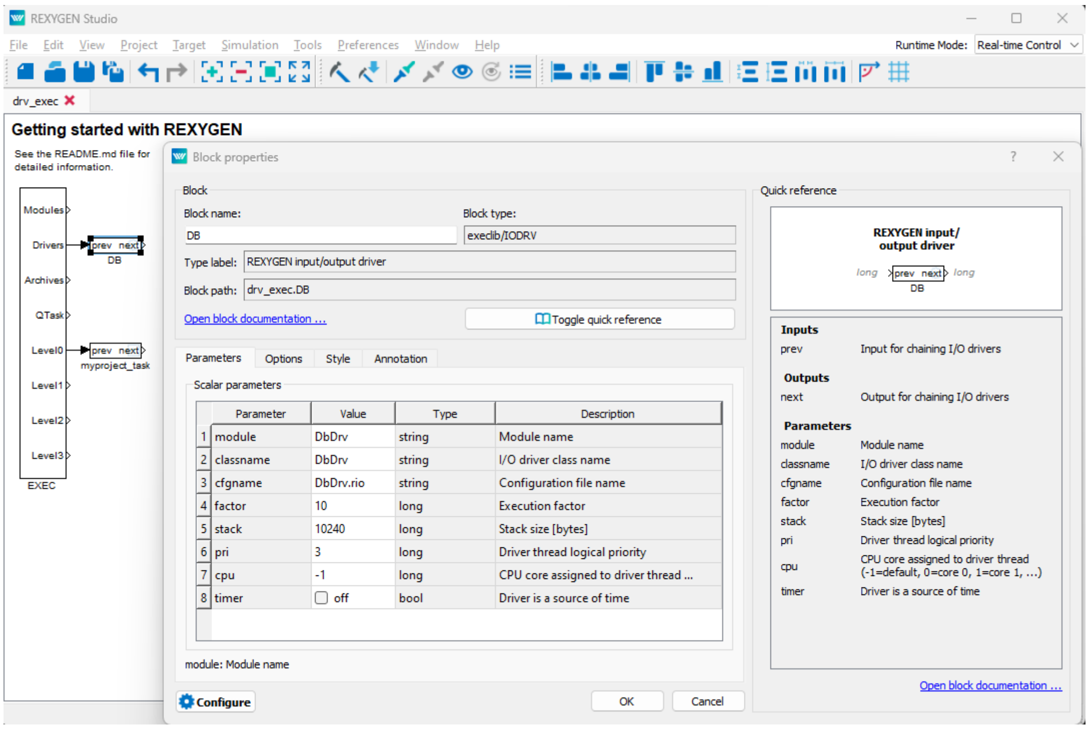
Chapter 2

Including the Driver in the Project

The driver is included in the project as soon as it is added to the project's main file and its inputs and outputs are connected in the control algorithms.

2.1 Adding the `DbDrv` Driver

The project's main file with the `DbDrv` driver included is shown in Figure 2.1.


REXYGEN Studio
File Edit View Project Target Simulation Tools Preferences Window Help
Runtime Mode: Real-time Control
drv_exec
Getting started with REXYGEN
See the README.md file for detailed information.
Modules
Drivers
Archives
QTask
Level0
Level1
Level2
Level3
EXEC
prev next
DB
prev next
myproject_task

Block properties

Block

Block name: DB

Block type: execlib/IODRV

Type label: REXYGEN input/output driver

Block path: drv_exec.DB

Open block documentation ...

Toggle quick reference

Parameters | Options | Style | Annotation

Scalar parameters

	Parameter	Value	Type	Description
1	module	DbDrv	string	Module name
2	classname	DbDrv	string	I/O driver class name
3	cfgname	DbDrv.rio	string	Configuration file name
4	factor	10	long	Execution factor
5	stack	10240	long	Stack size [bytes]
6	pri	3	long	Driver thread logical priority
7	cpu	-1	long	CPU core assigned to driver thread ...
8	timer	☐ off	bool	Driver is a source of time

module: Module name

Configure | OK | Cancel

Quick reference

REXYGEN input/output driver

long prev next long
DB

Inputs

prev	Input for chaining I/O drivers

Outputs

next	Output for chaining I/O drivers

Parameters

module	Module name
classname	I/O driver class name
cfgname	Configuration file name
factor	Execution factor
stack	Stack size [bytes]
pri	Driver thread logical priority
cpu	CPU core assigned to driver thread (-1=default, 0=core 0, 1=core 1, ...)
timer	Driver is a source of time

Open block documentation ...

Figure 2.1: An example of the project main file with the `DbDrv` driver included

To include the driver in the project, add a block of type `IODRV`, renamed to `DB`, and connect it to the `Drivers` output of the main `EXEC` block. The name of this block (`DB`, see Fig. 2.1), is the prefix of all input and output signals provided by this driver. The four most important parameters are:

`module` – name of the module linked to the driver, `DbDrv` in this case.

`classname` – class of the driver, `DbDrv` for the respective driver.

`cfgname` – name of the driver configuration file, e.g., `DBdrv.rio`.

`factor` – multiple of the EXEC block's tick parameter defining the driver's task execution period.

These parameters of the `IODRV` function block are configured in the REXYGEN Studio program. The configuration dialog is also shown in Fig. 2.1.

The `Configure` button opens the configuration dialog of the `DbDrv` driver, which is described in chapter 3.

2.2 Connecting the Inputs and Outputs in the Control Algorithm

The inputs and outputs of the driver must be interconnected with the individual tasks (`.mdl` files). The individual tasks (`QTASK` or `TASK` blocks) are connected to the `QTask`, `Level0`,..., `Level3` outputs of the main `EXEC` block. Use the blocks depicted in Fig. 2.2 within the individual tasks to interchange data between the control algorithm and the `DbDrv` driver.

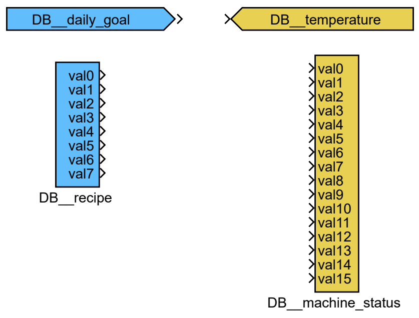


Figure 2.2: Example of input and output flags of the `DbDrv` driver

The `From` block, which allows the user to read one input signal, has the `Goto tag` set to `DB__<IN>`. The `Goto` block, which allows the user to set one output signal, has the `Goto tag` set to `DB__<OUT>`, where `<IN>` and `<OUT>` are strings referring to the items defined in the `*.rio` configuration file. The blocks always have the `DB` prefix right at the beginning of the tag followed by two underscores. The blocks with multiple inputs/outputs have this prefix directly in their name.

The use of multi-input/output blocks is recommended if data exchange rate (sampling frequency) is the priority. See the function block reference manual [2] for details about INOCT, OUTOCT, INHEXD, OUTHEXD blocks.

Each I/O name must be unique in all groups. Therefore, the section Group has an optional parameter Name (see chapter 3 for details about the Group section). In this case, an I/O signal is referenced by <group_name>_<item_name>.

The installation of the REXYGEN system includes a library of examples, where, among other things, the section 0404-01_DbDrv is dedicated to the use of the DbDrv driver. The example 0404-01-00_IO_Flags contains a library of usable inputs and outputs.

Chapter 3

I/O Driver Configuration

This chapter describes the configuration of individual input and output signals and their symbolic naming. The signals are mapped to the corresponding database.

The configuration dialog is part of the `DbDrv_H.dll`. It can be activated from REXYGEN Studio by pressing the `Configure` button in the parameters dialog of the `IODRV` block (see chapter 2). The resulting configuration is stored in a `*.rio` file, standard for other REXYGEN drivers. The dialog is segmented into three tabs, each detailed below.

3.1 Connection Tab

The first tab is dedicated to configuring the connection parameters for establishing a link with the database. This tab is illustrated in Figure 3.1. Below are explanations for each of the individual parameters:

`Connection type` – The so-called *connection-string* defining the database to connect to. A full *connection-string* with all the parameters can be used. Alternatively, it is possible to define the connection by DSN (DataSourceName), which is defined within the ODBC interface and contains all the necessary information for the connection.

`Driver` – The ODBC connection name for the database, such as `MySQL`, `MariaDB`, `MSSQL` or `PostgreSQL`.

`Server host` – Address of the server where the database is hosted.

`Server port` – Port number used for the database server connection.

`Database` – Name of the database to connect to.

`User (UID)` – Username associated with the database account.

`Password (PWD)` – Corresponding password for the database user.

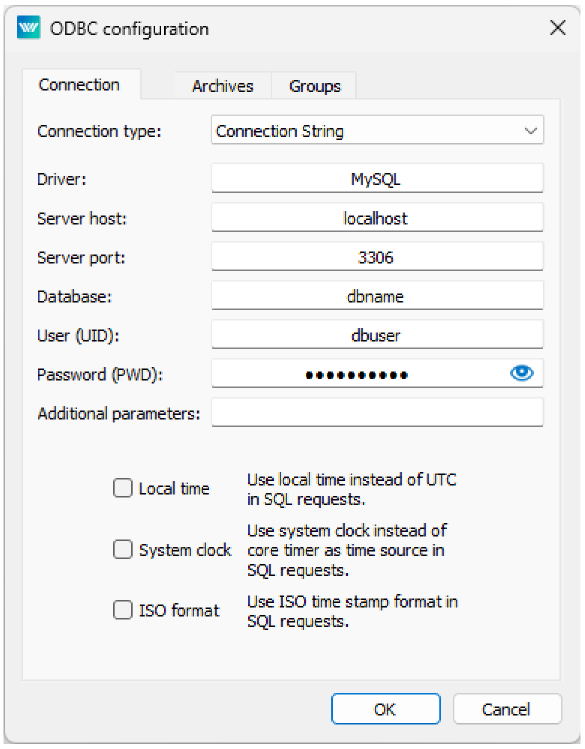
`Additional parameters` – Extra parameters or settings required for specific database connections.

In addition to the above parameters, the following checkboxes are available:

`Local time` – Use local time instead of UTC in SQL requests.

`System clock` – Use system clock instead of RexCore timer as time source in SQL requests. See `TRND` block documentation for more information about `SYSCLOCK` and `CORETIMER`.

`ISO format` – Use ISO time stamp format in SQL requests.



Connection type: Connection String
Driver: MySQL
Server host: localhost
Server port: 3306
Database: dbname
User (UID): dbuser
Password (PWD): ●●●●●●●●●●
Additional parameters:

☐ Local time — Use local time instead of UTC in SQL requests.
☐ System clock — Use system clock instead of core timer as time source in SQL requests.
☐ ISO format — Use ISO time stamp format in SQL requests.

Figure 3.1: Configuration dialog of `DbDrv` driver – Database connection details

3.2 Archives tab

This tab can be used to configure archiving from RexCore to the database. The tab's appearance is shown in Figure 3.2, and the meaning of its items is as follows:

`Mode` – Defines the structure of exporting data to the database. The options are:

0 Nothing gets exported (used for disabling the item temporarily).

1 Only the alarms and events are exported (filtered by additional parameters). The table in the database must contain the following columns: `Time`, `AlarmID`, `Code`, `Level`, `Value`.

2 Only the trends are exported, i.e. the data stored by the `TRND` block. The data is filtered by additional parameters. The table in the database must contain the following columns: `Time`, `GroupID`, `Value1`, `Value2`, ...

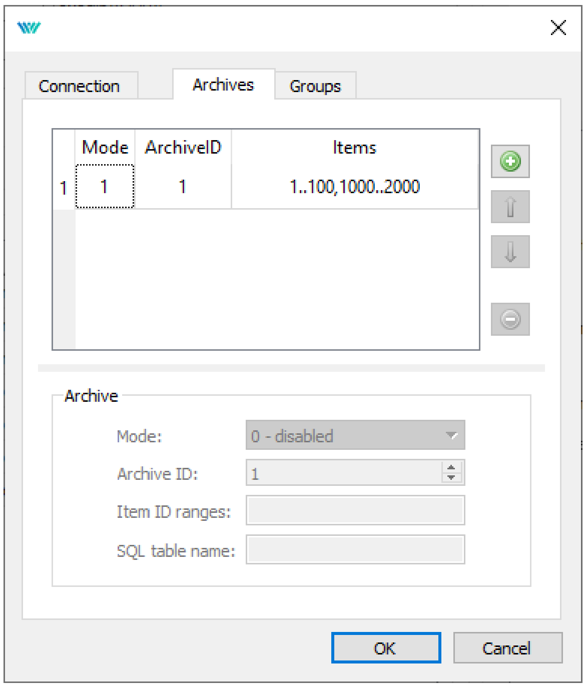
3 Only the trends are exported, but on the contrary to the above the `SQL` parameter has the meaning of a full SQL query, to which the values are injected. The following placeholders can be used: ?G = value of parametr GlobalId in *.rio file; ?S = value of parametr GlobalString in *.rio file; ?Y = year; ?M = month; ?D = day of month; ?T = timestamp (time and date of archive item); ?N = nanoseconds (in second) of timestamp; ?C = code; ?L = level; ?I =itemID; ?1 =1st item; ?2 =2nd item; ... A plain question mark has the meaning of *next item* in the following order: 1st value, 2nd value, ...;). Additional `SQL2` parameter has the meaning of an SQL query used for detection of last record stored in the database which is used for automatic loading of data archived when the database was not available.

4 Only the alarms are exported and the `SQL` parameter has the meaning of a full SQL query. The placeholders and the `SQL` parameter are like in previous mode.

`Archive ID` – Number of archive to read the data from. The archives are numbered from 1 in the order of appearance in the configuration of REXYGEN executive (`ARCHIVE` blocks connected to the `EXEC` block).

`Item ID ranges` – A range of IDs (the `id` parameter of the originating block) to export from archive to the database (in the `Archive` section).There must be even number of entries, where the odd entries define the start of an interval and the even entries define its end. Therefore e.g. "100,100,104,109" means IDs 100, 104, 105, . . . , 109. The entries must be sorted in ascending order.

`SQL table name` – Table name or full SQL command as defined by the `Mode` parameter. In some cases the notation `<database_name>.<table_name>` must be used for referencing database tables.



	Mode	ArchiveID	Items
1	1	1	1..100,1000..2000

Archive

Mode: 0 - disabled

Archive ID: 1

Item ID ranges:

SQL table name:

Figure 3.2: Configuration dialog of `DbDrv` driver – Archives section configuration

3.3 Groups tab

The final tab is used for directly reading and writing data to and from the algorithm. The Groups tab is depicted in Figure 3.3 and is subdivided into three sections:

- `Group table` – This section displays all the prepared data groups. Meaning of the columns are as follows:

  `Name` – Name of the group for reading/writing items.

  `Mode` – The available modes are described in the table below.

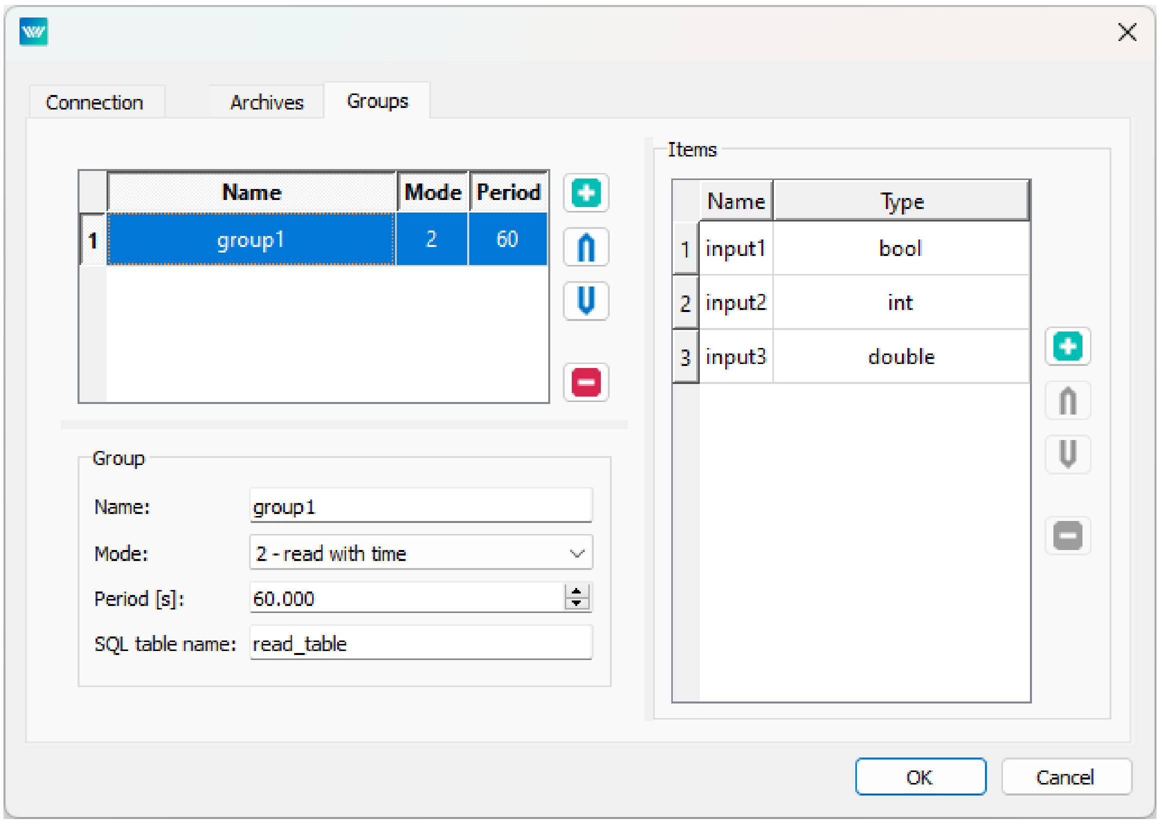
  `Period [s]` – Period in seconds to generate the SQL query.

- `Group settings` – This section encompasses the configurations for the presently selected group. Within this section, you will find the three aforementioned items, along with an additional item, either `SQL table name` or `SQL query`, contingent upon the selected `Mode`. In some cases the notation `<database_name>.<table_name>` must be used for referencing database tables.

- `Item table` – This table is only shown when a group is selected, displaying the reading and writing items for that group. Columns have following meaning:

  `Name` – Name of the reading/writing item.

Type – The possible types are: bool, int, double, string, large.



	Name	Mode	Period
1	group1	2	60

Group

Name: group1

Mode: 2 - read with time

Period [s]: 60.000

SQL table name: read_table

	Name	Type
1	input1	bool
2	input2	int
3	input3	double

Figure 3.3: Configuration dialog of DbDrv driver – Groups section configuration

	The Available Group Modes
0	Nothing gets read (used for disabling the item temporarily).
1	It is assumed that the table is ordered by the ID column. The row with the highest ID is supplied to the corresponding input flags in the task. The columns and the items/flags must have the same name.
2	It is assumed that the table is ordered by the Time column (ID is the secondary key). The row with the highest time not placed in the future is selected and the resulting data is supplied to the corresponding input flags in the task. The columns and the items/flags must have the same name. This mode allows applying a pre-generated sequence of data.
3	The SQL query from the SQL parameter is executed, the inputs are updated by the first row of the response (1st column to the 1st item, ...). It is possible to use ?T in the SQL query, which gets replaced by the current time. It is also possible to use ?1 in the SQL query, which gets replaced by the value of 1st item, ?2 , which gets replaced by the value of 2st item,...

128	Nothing gets written (used for disabling the item temporarily).
129	The values from the corresponding flags in the tasks are written to the database. The columns and the items/flags must have the same name.
130	Similar to the above, only there is one more column named `Time` which contains the current time (including date) of the REXYGEN runtime core in UTC (or different time defined by Options-parameter).
131	The SQL query from the `SQL` parameter is executed. The following placeholders can be used: ?G = value of parametr GlobalId in *.rio file; ?S = value of parametr GlobalString in *.rio file; ?Y = actual year; ?M = actual month; ?D = actual day of month; ?T = current time 9including date); ?I =itemID; ?1 =1st item; ?2 =2nd item; ... A plain question mark has the meaning of *next item* in the following order: 1st value, 2nd value, ...;)

3.4 Special signals

There are additional auxiliary signals for each I/O signal. These can be assessed by appending the following strings to the signal reference:

`_Status` – status code - result of last query. Codes are: 0 ... last request succeeded 1 ... last request failed 2 ... no request perform yet 3 ... last request return empty dataset 101 ... last request failed due database disconnected 102 ... connection to database is not established, no request required

`_Disable` – If `True`, the read/write operations for the whole group are disabled.

`_Trigger` – A rising edge triggers execution of the read/write operation.

`_Age` – Number of seconds since the last read/write database access.

`_Fresh` – Same as `_Age`

`_AgeDb` – Number of seconds since the last read/write database access, on the contrary to the above the age is defined by the database item.

`_FreshDb` – Same as `_AgeDb`

`_Period` – Contains or sets the `Period` parameter, i.e. the period of SQL query execution as defined for each group.

`_Done` – indicate last request finished successfully (value is set to off after read if Trigger is off)

`_Error` – indicate last request failed (value is set to off after read if Trigger is off)

`_Empty` – indicate last request return empty dataset (value is set to off after read if Trigger is off)

This attribute are attribute of group, because whole group is written/read by one SQL command. Therefore attribute can be accessed by signal reference string `DB__<group_name>_<attribute_name>`, where `<group_name>` is value of optional parameter `Name` in section `Group`.

There are additional auxiliary global signals:

- `Connect` – connection to database server.
- `Connected` – status of the connection to database server.
- `Reset` – resetting archive reading (all archiving groups).
- `Resetting` – pushed off when resetting is finished.
- `GlobalId` – set or get value of GlobalID parameter in *.rio file.
- `GlobalString` – set or get value of GlobalString parameter in *.rio file.

There are additional auxiliary signals for each archive worker (associated with Archive section in configuration file). It require optional parameter `Name` in `Archive` section. These can be accessed by signal reference string `DB__<archive_name>_<option_name>`, where `<option_name>` is:

- `_Count` – number of successfully written archive items into database
- `_ErrorCount` – number of failed archive items (skipped and not written into database)
- `_LastErrorStr` – timestamp of last processed archive item (written into database or skipped)

Chapter 4

Implementation Details

Additional information about the use and implementation of the `DbDrv` driver in the REXYGEN system is gathered in this chapter.

- The `Items` parameter in the `Archive` section is a list of numbers, where the odd entries mean *from* and the even ones *to*. E.g., `Items "2, 5, 10, 15"` exports items with IDs 2 to 5 and 10 to 15. There must be an even number of entries, even if exporting items with only one ID. The entries in the `Items` parameter must be sorted in ascending order.

- After connection to the database is established, data configured by `Archive` sections are automatically loaded to the database starting from the last record stored. This feature is not active when the database table is empty. In such a case, data loading starts from the current time only. If you want to load old data from archives to an empty database, insert a dummy record with a timestamp before the point you want to start from. Loading of old data is limited to 100 records per driver execution period to avoid performance issues. Please keep this in mind when setting the execution period.

- Although the majority of database systems are case-insensitive, the control system REXYGEN is case-sensitive. Therefore, the `DbDrv` driver is also case-sensitive in the I/O flags (the flags correspond with column names in the database).

- All values written or read to/from the database are decimal numbers (type double). The database columns can be of other types because SQL queries are textual. The optional parameter `Type` can be used, where `i` means the value is processed as `long` type, `b` denotes a `bool` type, `s` denotes a `string` type, and `r` denotes a `real`. For example, `Type "rrisb"` means the 1st and 2nd values are real numbers, the 3rd value is an integer, the 4th value is a string, and the 5th value is a boolean.

- The flags must be unique in the whole project because they contain no `Group` identifier. Only the first occurrence is processed in the case of duplicities.

- It is possible to define an (optional) parameter `Name` in the `Group` section. The flags must be in the form `<group_name>_<item_name>` in this case.
- The driver needs a username and password for login into the database. Both are stored in the *.rio file in plain text as all other parameters. Therefore, it is strongly recommended to use a dedicated login name with very restricted permissions.
- All timestamps (e.g., substitute for `?T` in SQL queries) are expanded into a string in the form `<year>-<month>-<day> <hour>:<minute>:<second>.<microsecond>`. The UTC timezone is used (if not defined by another timezone by the Options-parameter).
- Using an SQL data type with at least microsecond resolution is recommended for timestamps.
  - MySQL: `DATETIME(6)` [1]
  - Microsoft SQL (2008+): `datetime2` [2]
  - PostgreSQL: `timestamp` [3]
- Current implementation limits the SQL string to 1023 characters (after expansion of question marks). The parameter `Items` (in both `Archive` and `Group` sections) is limited to 64 values.
- The column `Code` in alarm export is integer number where lowest 5bit is alarm class and higer 3 bits is alarm subtype. The classes are:

[1] http://dev.mysql.com/doc/refman/5.7/en/fractional-seconds.html
[2] https://msdn.microsoft.com/en-us/library/bb677335.aspx
[3] https://www.postgresql.org/docs/current/static/datatype-datetime.html

0	System alarm
1	Bool alarm
2	Byte value alarm
3	Short value alarm (signed 16 bits integer number)
4	Long value alarm (signed 32 bits integer number)
5	Word value alarm (unsigned 16 bits integer number)
6	DWord value alarm (unsigned 32 bits integer number)
7	Float value alarm
8	Double value alarm
10	Large value alarm (signed 64 bits integer number)
12	String value alarm
13 .. 16	Not used
17	Bool value group event
18	Byte value group event
19	Short value group event
20	Long value group event
21	Word value group event
22	Dword value group event
23	Float value group event
24	Double value group event
26	Large value group event
31	Alarm acknowledge

The subtypes for system alarms are:

0	Date mark (is not exported)
1	Executive event
2	Archive event

The level indicate event in this case. The executive events are:

0	System reset
1	Download begin
2	Download end
3	Download failed
4	Executive stop
5	Executive start
6	Executive swap
7	Time set

and the archive event:

0	Archive cleared (not used now)
1	Archive reconstruction saved (not used now)
2	Archive reconstruction normal (not used now)
3	Checksum error (not used now)
4	Integrity error (not used now)
5	Sizes changed (not used now)
6	Limit exceed (disk archives only)
7	Buffer overflow

The subtypes for boolean alarms are:

0	High to Low (e.g. the attached boolean variable have been changed from the high/true/1 value to the low/false/0 value)
1	Low to High (e.g. the attached boolean variable have been changed from the low/false/0 value to the high/true/1 value)

The subtypes for numeric alarms are:

0	low alarm
1	high alarm
2	2nd low alarm
3	2nd high alarm

The level 0 indicate end of alarm conditions. The alarms with level from 128 to 255 not indicate end of alarm conditions by this special alarm event. The alarm acknowledge should has same subtype and level as acknowledged alarm. However not active alarms are regarded as acknowledged by the acknowledge of any subtype. The 1st level alarm is also regarded as acknowledge by the acknowledge of the 2nd level.

Chapter 5

Troubleshooting

First and foremost, it's advisable to explore the library of examples, especially the section `0404-01_DbDrv`, which pertain to the usage of `DbDrv`.

Just like in the case of any other problem, it is recommended to view the error and debug information (the `System Log` section in REXYGEN Studio). Unsuccessful database connections, misconfigured SQL queries, and other related issues are listed in the log. The most frequent problems include:

- The ODBC interface for the corresponding database is not installed or correctly configured on the device (the odbcinst.ini and odbc.ini files in Linux).
- Invalid database *connection-string*.
- The requested tables are not available in the database. Remember, the database might be case-sensitive.
- Inconsistency in naming of the columns. Again, the database might be case-sensitive.
- Although the `DbDrv` driver uses very simple SQL syntax, there are some differences among individual database systems.
- Especially when defining the SQL queries by hand, it is necessary to double-check the syntax.
- Duplicated item name in the `Items` parameter, which results in the item being not available.

In the case that the given input or output works with other software tools and does not work in REXYGEN, please report the problem to us. E-mail is preferred; reach us at `support@rexygen.com`. Please include the following information in your description to help us process your request as soon as possible:

- Identification of the REXYGEN system you are using. Simply export it to a file using the REXYGEN Studio (Target → Licensing → Export).

- Short and accurate description of your problem.
- The configuration files of REXYGEN (`.mdl` files) reduced to the simplest case which still demonstrates the problematic behavior.

Bibliography

[1] REX Controls s.r.o.. *Getting started with REXYGEN on Debian*, 2020. →.

[2] REX Controls s.r.o.. *Function blocks of REXYGEN – reference manual*, 2020. →.

Documentation reference number: 16843