UC20-SL2000-OLAC

2638920000

Quick Start Guide saving plc variables on SD-card

Using recipe management to save IEC 61131-3 variables to a file and mount the SD-card

QSG0016v03-UC20

Revision history

Version	Date	Change log
01	2019-11	First released version
02	2020-07	Revision for web launch
03	2020-10	Mounting SD-Card is not necessary since u-create studio 1.20

Contact

Weidmüller Interface GmbH & Co. KG
Klingenbergstraße 26
32758 Detmold, Germany
T +49 5231 14-0
F +49 5231 14-292083
info@weidmueller.com
www.weidmueller.com

For further support please contact your local sales representative.

Author: w010485

1. Content

2. Warning and disclaimer

Warning
Controls may fail in unsafe operating conditions, causing uncontrolled operation of the controlled devices. Such hazardous events can result in death and / or serious injury and / or property damage. Therefore, there must be provide safety equipment/ electrical safety design or other redundant safety features that are independent from the automation system.

Disclaimer
This Example / Application Note does not relieve you of the obligation to handle it safely during use, installation, operation and maintenance. Each user is responsible for the correct operation of his control system.
By using this program example / application note prepared by Weidmüller, you accept that Weidmüller cannot be held liable for any damage to property and / or personal injury that may occur because of the use.

Note
The application examples do not represent customer-specific solutions, they are simply intended to help for typical tasks. The user is responsible for the proper operation of the described products. This application example does not relieve you of the obligation of safe use, installation, operation and maintenance. Application examples are not binding and do not claim to be complete in terms of configuration as well as any contingencies.
By using this Application Example, you acknowledge that we cannot be held liable for any damages beyond the described liability regime. We reserve the right to make changes to this sample application at any time without notice.
In case of discrepancies between the proposals in the application example and other Weidmüller publications, like manuals, such contents always have more priority to the examples.
We assume no liability for the information contained in this document. Our liability, for whatever legal reason, for damages caused by the use of the examples, instructions, programs, project planning and performance data, etc. described in this application example is excluded.

Security notes
In order to protect equipment, systems, machines and networks against cyber threats, it is necessary to implement (and maintain) a complete state-of-the-art industrial security concept. The customer is responsible for preventing unauthorized access to his equipment, systems, machines and networks. Systems, machines and components should only be connected to the corporate network or the Internet if necessary and appropriate safeguards (such as firewalls and network segmentation) have been taken.

3. Abstract

The scope of this document is to show how to store IEC 61131-3 variables on the SD-card and load them after a reboot of the controller. Further, the document explains how a SD-card can be mounted from out of the IEC 61131-3 runtime.

WARNING: Writing data to a filesystem reduces the lifespan of the used memory, do not use the internal flash of the controller and try to limit the write cycles to the used storage medium!

To understand the content of this document, basic knowledge about handling projects in u-create studio is required. All programming snippets are written in structured text.

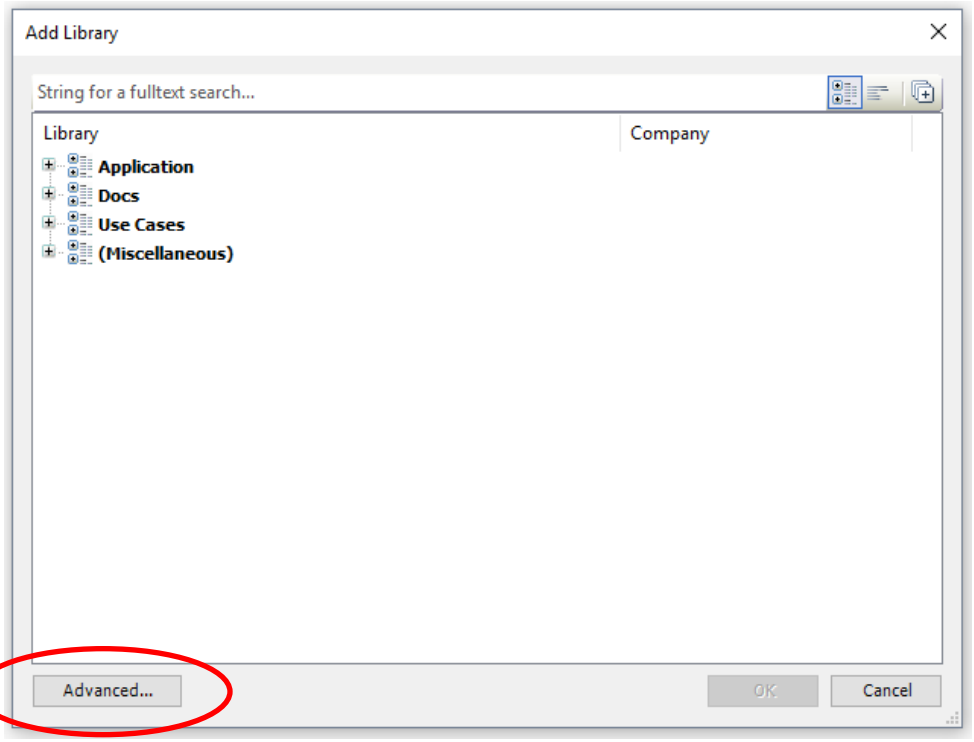
4. Mounting the SD-card

It is highly recommended to use a SD-card, because this makes sure that the controller itself is not damaged during operation. This storage medium is not mounted by default, it is necessary to mount it manually. The following section shows how this can be achieved from out of the IEC 61131-3 environment.

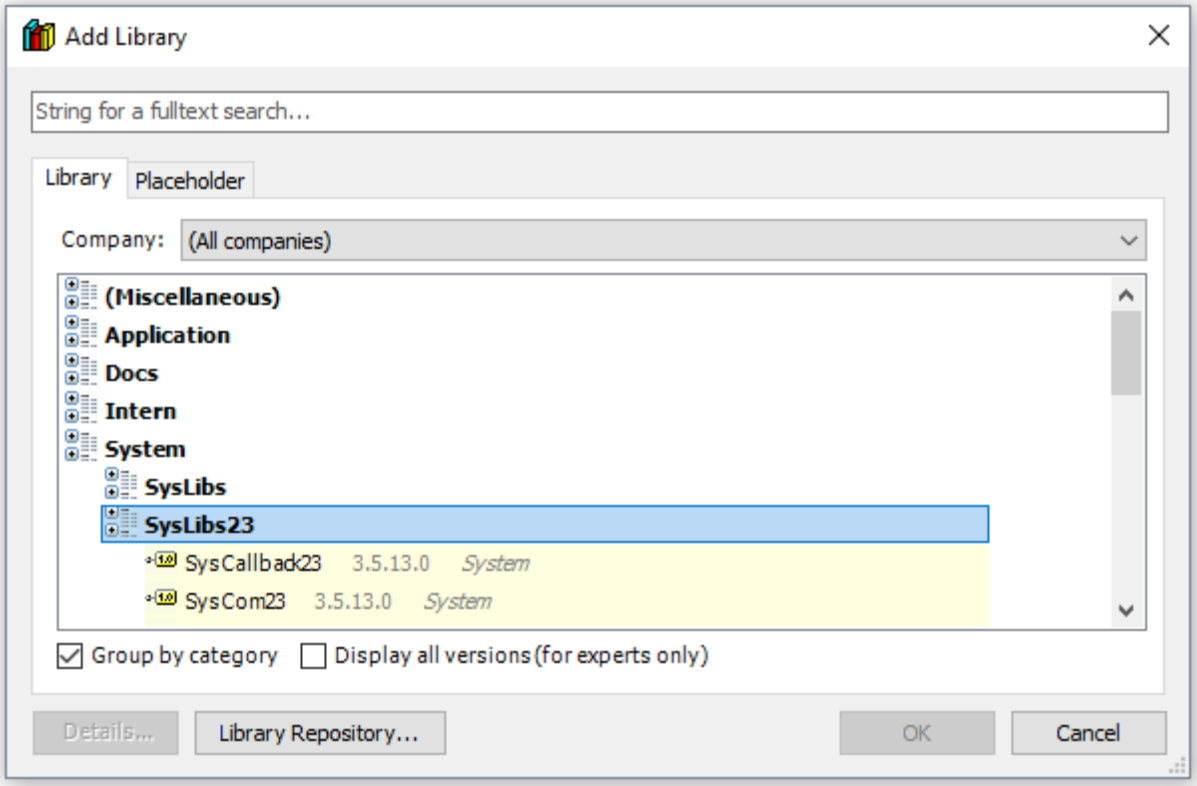
Since Version 1.20 of u-create studio, this step is obsolete, as the SD-Card is now premounted! The path is: /media/sd0/

4.1. Used libraries

The SD-card is mounted by using shell commands. These commands are executed by functions of the library “**SysOS23**”.



To include the library, the **Add Library** window of the **Library Manager** needs to be switched to the **Advanced** mode.



The library can be found under the path **System/SysLibs23/SysOS23.**

4.2. Executing shell command to mount the SD-card

After SysOS23 has been included, the function **SysExecuteCommand(STRING)** is available. The function executes shell commands that are inserted as string.

WARNING: Shell commands can affect the real-time of the OS. Controlled equipment should be in a save state before executing shall commands on a PLC!

```
(* prepare everything and read the data from the SD - Card for the fist time *)
IF ( xMountAndInit ) THEN
    (* create the directory to mount the SD - Card *)
    SysOS23.SysExecuteCommand('mkdir /home/admin/SD');
    (* mount the SD - Card to the created directory *)
    SysOS23.SysExecuteCommand('sudo mount /dev/mmcblk0p1 /home/admin/SD');
    (* do the initial read of the data *)
    BPApiRecipeRead('Peristent.txt', '/home/admin/SD');
    (* reset int bit *)
    xMountAndInit := FALSE;
END_IF
```

Mounting the SD-card is done in two steps. At first a **new directory is created,** to that the card is mounted later. Creating a new directory is done by the shell command **mkdir** followed by the

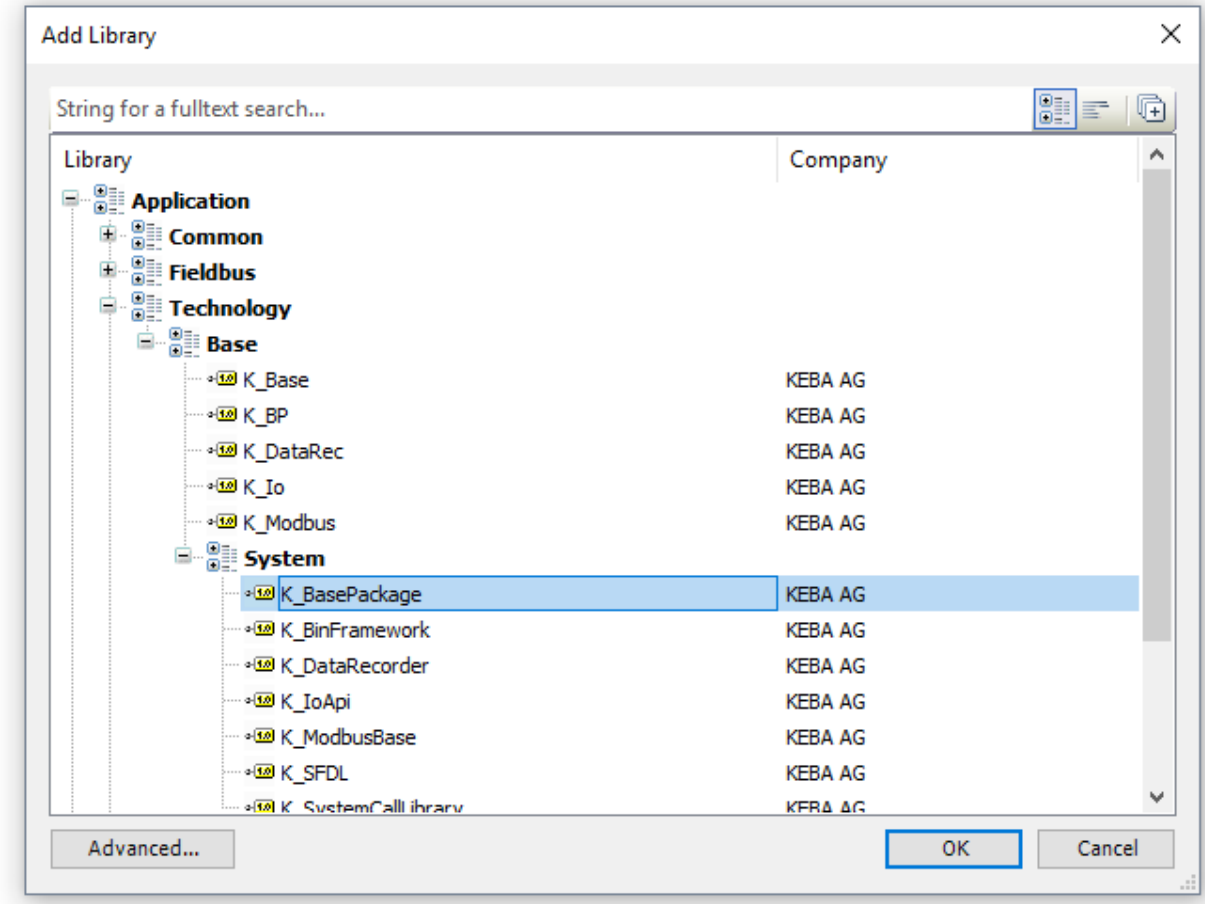
path of the new directory. The second step is to **mount** the card to the **created directory**. The command to mount devices is **mount**, followed by the device path **(/dev/mmcblk0p1)** and the directory path, where the device is mounted to.

5. Using recipe management to save and load variables

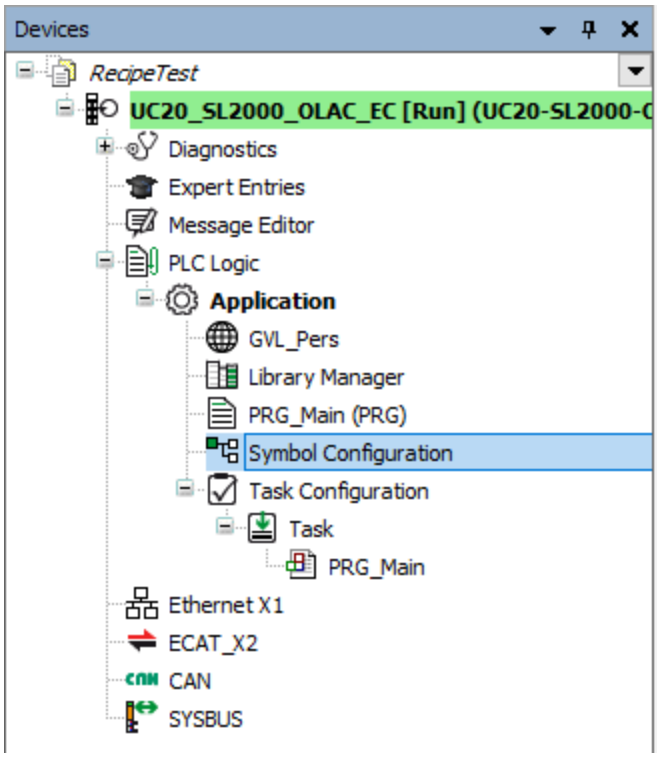
Recipes are files in the controllers file system that include a set of variables that can be stored and loaded by a function from the IEC 61131-3 runtime.

5.1. Preparing the plc project and the controller to work with recipes

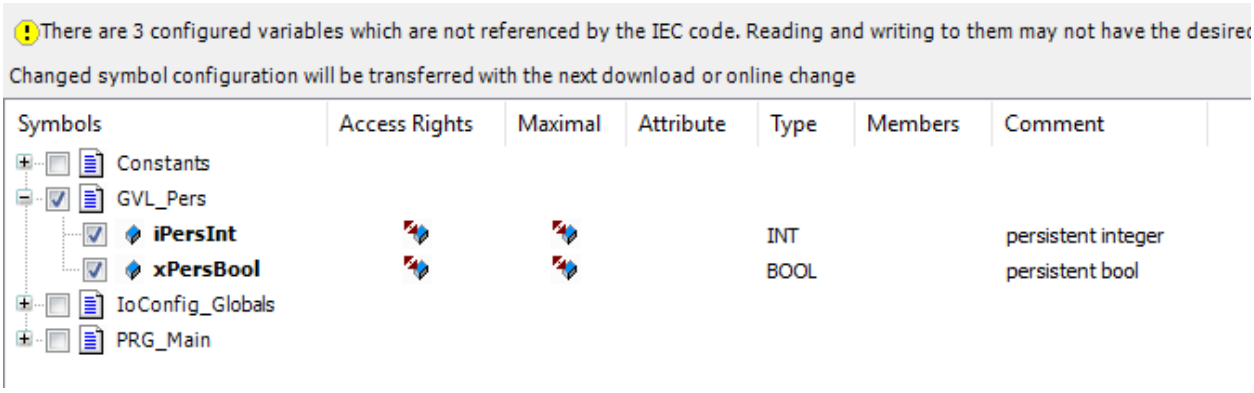
The first step is to include the **library K_BasePackage**. This library can be found in the library manager under **Application/Technology/Base/System/K_BasePackage**



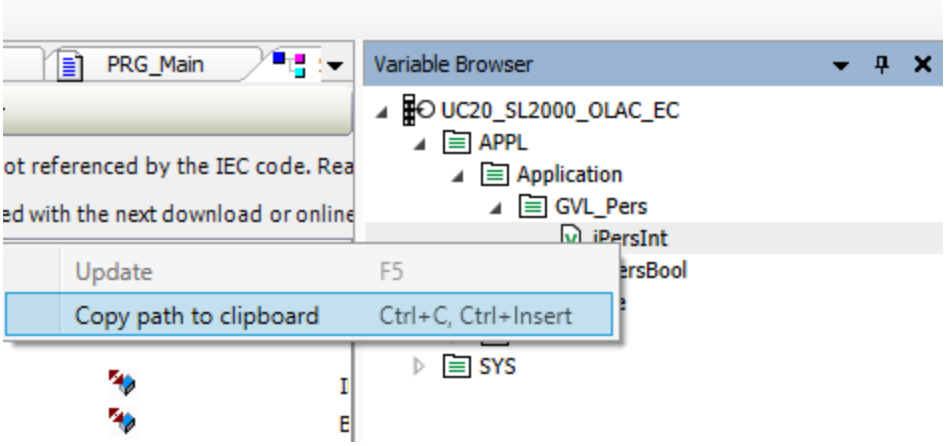
The next step is to add the **variables**, that shall be **saved persistently**, to the **symbol configuration**. If there is no symbol configuration in the current project, it is required to create one.



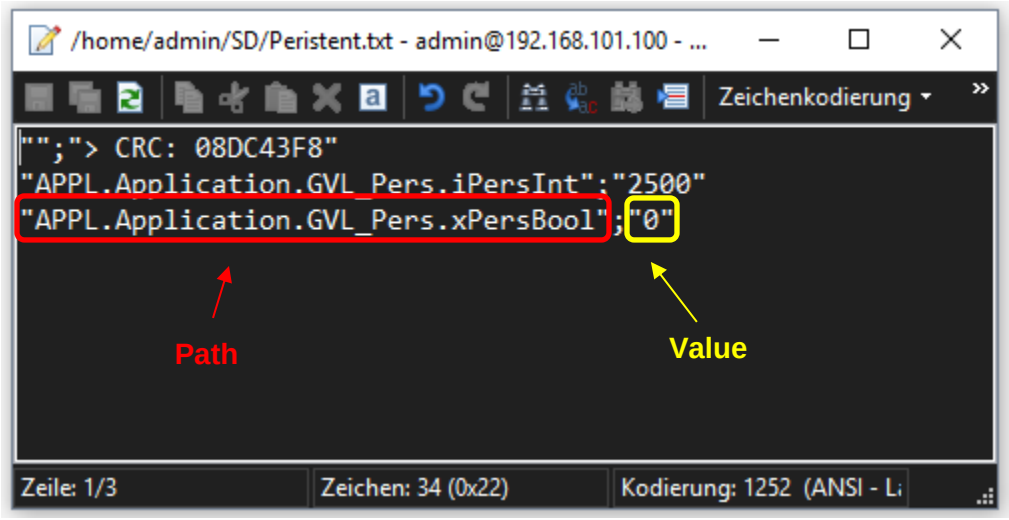
Once the symbol configuration is available in the project, different variables can be added. In this example, there is a global variable list with the name “GVL_Pers”. The two variables in this list are added to the symbol configuration and should be saved persistently.



Once the variables are added to the symbol configuration, they are available in the so called “**Variable Server**”.



The last step is to **add the path** of the variables from the **variable server** to the **recipe file**. The path can be copied once u-create studio is connected to the plc. Therefor the "**Variable Browser**" needs to be opened (**View/Variable Browser**). By a right click on the variable, it is possible to copy the file to the clipboard.



The recipe file is shown in the picture above. It needs to be created by the plc programmer and **stored on the SD-card**. The first line (the CRC-checksum) needs to be blank, if the file is created for the first time or the values were changed manually. It will be generated automatically by the recipe management after the first write cycle. The rest of the file is build up like this: The first value is the path of the variable on the variable server. The **semicolon separates** than the **value from the path.**

5.2. Reading and saving the recipe file from the IEC runtime

Once the preparations have been done, the recipe file can be used. This is done by two functions from the library **K_BasePackage**.

```
17    (* check if read is requested *)
18    IF ( xReadFALSE ) THEN
19        (* execute recipe read function *)
20        BPApiRecipeRead('Peristent.txt', '/home/admin/SD');
21        (* rest read bit *)
22        xReadFALSE := FALSE;
23    END_IF
```

The function to read Data from the file is called “**BPApiRecipeRead(STRING, STRING)**”. It requires two inputs. The first is the name of the recipe file and the second one is the path to the folder, where the recipe file is stored. The path is the directory of the SD-card that is mounted (see 4.2). The data is written to the variables, defined in the recipe file, during function call.

```
25    (* check if save is requested *)
26    IF ( xSaveFALSE ) THEN
27        (* execute recipe save function *)
28        BPApiRecipeUpdateFile('Peristent.txt', '/home/admin/SD');
29        (* reset save bit *)
30        xSaveFALSE := FALSE;
31    END_IFRETURN
```

The second function updates the recipe file, so it stores the variables to the SD-card. It is called “**BPApiRecipeUpdateFile(STRING, STRING)**”. Again, the first parameter is the file name and the second the path. Once the function is executed, all variables defined in the file are saved to the SD-card.