

APPLICATION EXAMPLE: AE-0565

# AC500 V3 FILE HANDLING

## WRITE, READ AND APPEND FILES



# Contents

<b>1 Disclaimer .....</b>	<b>3</b>
<b>2 Introduction .....</b>	<b>4</b>
2.1 Scope of the document .....	4
2.2 Compatibility .....	4
2.3 Overview .....	4
<b>3 CAA File library .....</b>	<b>5</b>
<b>4 Save a Snapshot .....</b>	<b>6</b>
4.1 Visualization .....	6
4.2 Program description .....	7
4.2.1 Analog signals .....	7
4.2.2 Snapshot .....	7
4.3 Usage in computer (Excel) .....	8
<b>5 Adding on event .....</b>	<b>9</b>
5.1 Visualization .....	9
5.2 Program description .....	10
5.3 Usage in computer (Excel) .....	10

# 1 Disclaimer

A. For customers domiciled outside Germany /

Für Kunden mit Sitz außerhalb Deutschlands

**„Warranty, Liability:**

The user shall be solely responsible for the use of this products described within this file. ABB shall be under no warranty whatsoever. ABB's liability in connection with application of the products or examples provided or the files included within these products, irrespective of the legal ground, shall be excluded. The exclusion of liability shall not apply in the case of intention or gross negligence. The present declaration shall be governed by and construed in accordance with the laws of Switzerland under exclusion of its conflict of laws rules and of the Vienna Convention on the International Sale of Goods (CISG)."

**„Gewährleistung und Haftung:**

Der Nutzer ist allein für die Verwendung des in diesem Dokument beschriebenen Produkte und beschriebenen Anwendungsbeispiele verantwortlich.

ABB unterliegt keiner Gewährleistung. Die Haftung von ABB im Zusammenhang mit diesem Anwendungsbeispiel oder den in dieser Datei enthaltenen Dateien - gleich aus welchem Rechtsgrund - ist ausgeschlossen. Dieser Ausschluss gilt nicht im Falle von Vorsatz oder grober Fahrlässigkeit. Diese Erklärung unterliegt Schweizer Recht unter Ausschluss der Verweisungsnormen und des UN-Kaufrechts (CISG)."

B. Nur für Kunden mit Sitz in Deutschland

**„Gewährleistung und Haftung:**

Die in diesem Dokument beschriebenen Anwendungsbeispiele oder enthaltenen Dateien beschreiben eine mögliche Anwendung der AC500 bzw. zeigen eine mögliche Einsatzart. Sie stellen nur Beispiele für Programmierungen dar, sind aber keine fertigen Lösungen. Eine Gewähr kann nicht übernommen werden.

Der Nutzer ist für die ordnungsgemäße, insbesondere vollständige und fehlerfreie Programmierung der Steuerungen selbst verantwortlich. Im Falle der teilweisen oder ganzen Übernahme der Programmierbeispiele können gegen ABB keine Ansprüche geltend gemacht werden.

Die Haftung von ABB, gleich aus welchem Rechtsgrund, im Zusammenhang mit den Anwendungsbeispielen oder den in dieser Datei enthaltenen Beschreibung wird ausgeschlossen. Der Haftungsausschluss gilt jedoch nicht in Fällen des Vorsatzes, der groben Fahrlässigkeit, bei Ansprüchen nach dem Produkthaftungsgesetz, im Falle der Verletzung des Lebens, des Körpers oder der Gesundheit oder bei schuldhafter Verletzung einer wesentlichen Vertragspflicht. Im Falle der Verletzung einer wesentlichen Vertragspflicht ist die Haftung jedoch auf den vertragstypischen, vorhersehbaren Schaden begrenzt, soweit nicht zugleich ein anderer der in Satz 2 dieses Unterabsatzes erwähnten Fälle gegeben ist. Eine Änderung der Beweislast zum Nachteil des Nutzers ist hiermit nicht verbunden.

Es gilt materielles deutsches Recht unter Ausschluss des UN-Kaufrechts."

## 2 Introduction

### 2.1 Scope of the document

This document gives an overview how to work with files in the AC500 V3. The library used for the file handling is the CAA File library.

Files can be used to store or read data on a hard drive. As disk either the internal userdisk or an external SD card can be used. The PM567X has a flashdisk in addition.

This example is working on the userdisk and the SD card.

There are two use cases described in this project.

The first one is to save a snapshot. The state of variables during a period of time is recorded. In this example analog input signals are written into a file. The content of the file can be read back into an array and be displayed in the visualization. By recording a new snapshot of the variables, the old one will be overwritten.

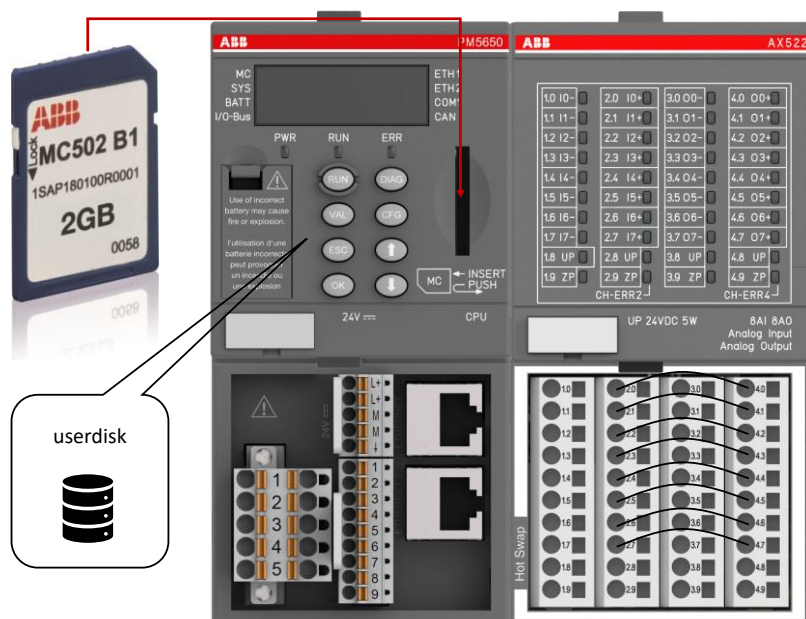
The other use case in this program is adding lines to a log file. Depending on the user input the current state of three variables is logged with a timestamp into a file. New values will be added to this file and the file will not be overwritten.

### 2.2 Compatibility

The application example explained in this document have been used with the below engineering system versions. They should also work with other versions, nevertheless some small adaptations may be necessary, for future versions.

- AC500 V3 PLC
- Automation Builder 2.6.0 or newer

### 2.3 Overview

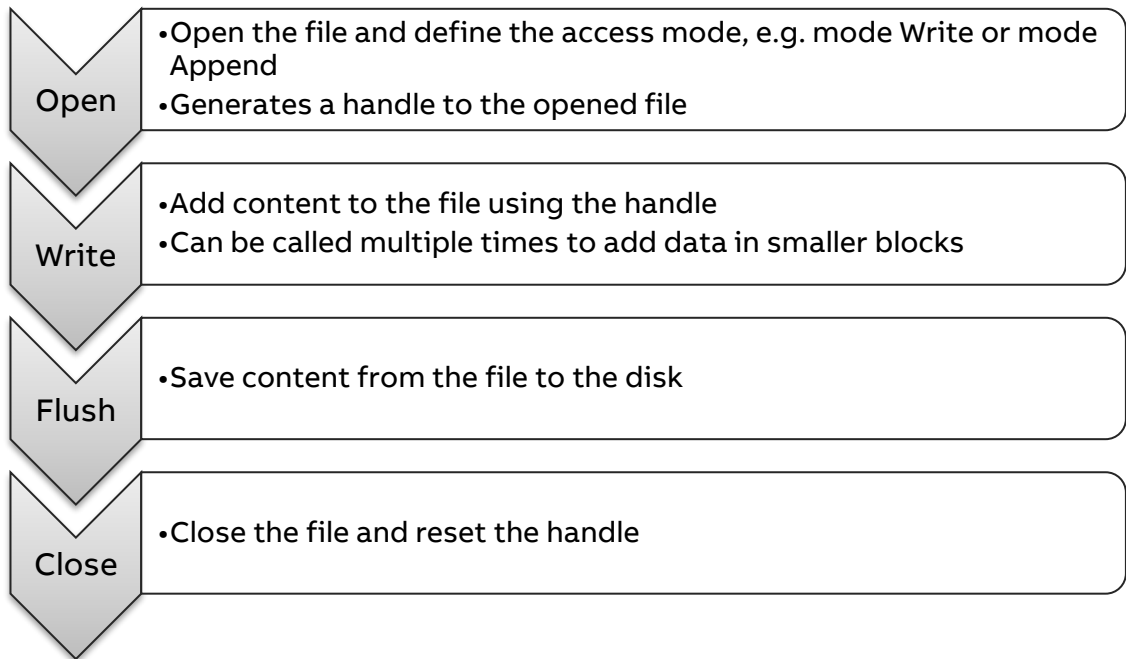


The SD card and the AX522 module are optional. The example is also working without them.

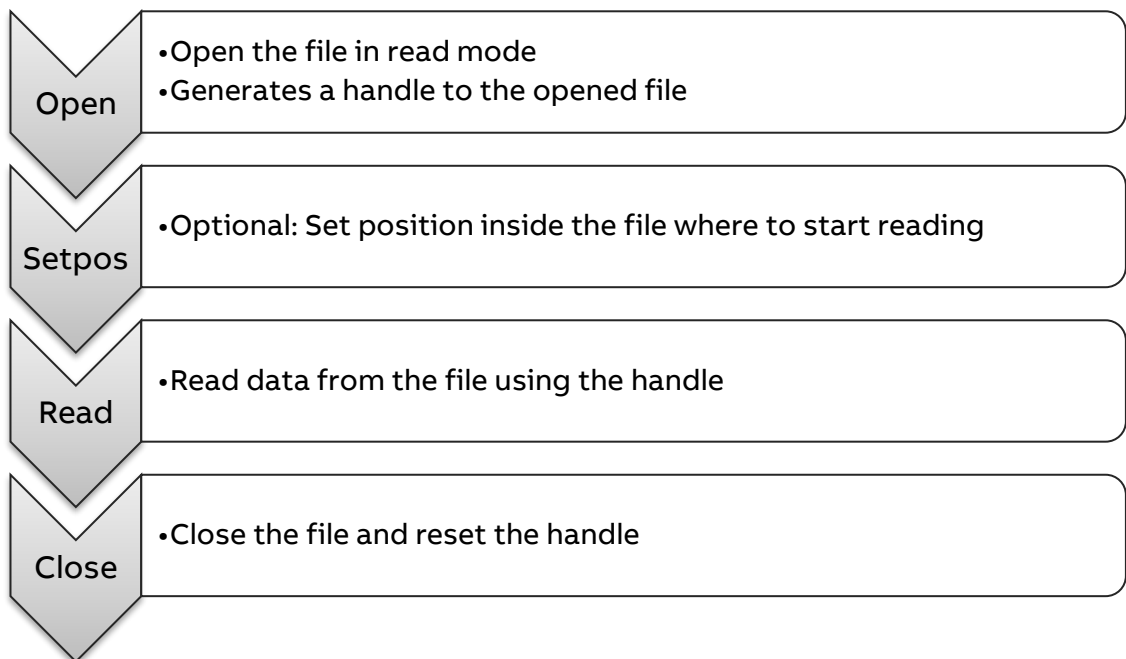
### 3 CAA File library

This Application example is based on the CAA File library version 3.5.17. Also with other library versions the functionality will work. The CAA File library can be used for directory handling and file handling. This example is only using the file functionality. A complete documentation about all function blocks is inside the library. For a file handling there are always several function blocks required. The processes below are showing the required function blocks for writing or reading files.

#### Writing files



#### Reading Files



## 4 Save a Snapshot

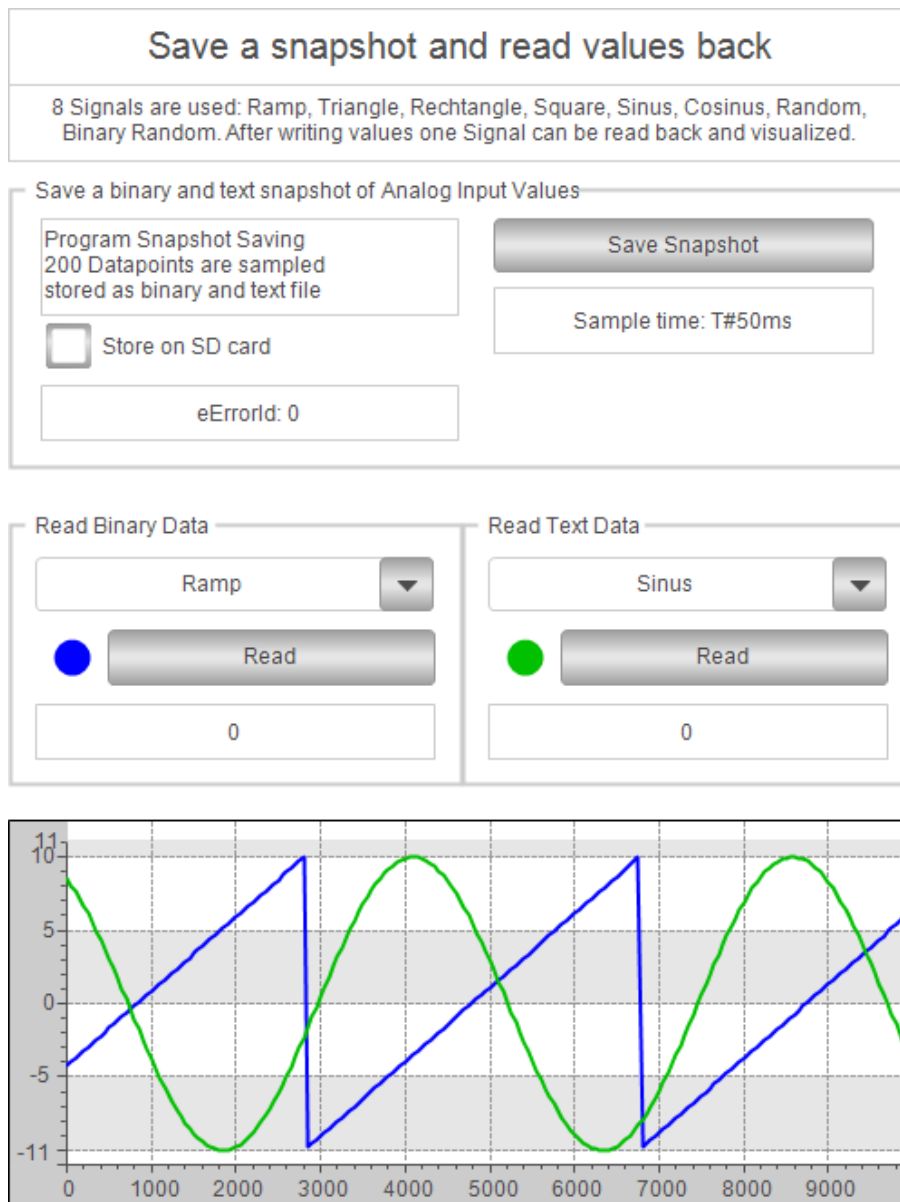
This program records the values of the analog inputs and logs them into a file. After saving the log file, the values can be read back and displayed to a graph. The values are saved in a binary format and in a readable CSV format. By default, the files are stored in the internal userdisk. The storage can also be changed to an SD card.

### 4.1 Visualization

On the left side of Main\_VISU the “Save a Snapshot” functionality is located.

By clicking “Save Snapshot” the program will record 200 samples of the eight analog input values with the specified sample time. Afterwards the values are written into a binary and a CSV file. In case of an error the Error ID is shown.

After successfully writing into the file, the user can select a signal which he wants to read from the binary data or from the text data (CSV-File). By clicking the read button, the selected signal will be displayed in blue for the binary data or in green for the text data. In case of an error the user gets the ID shown in the visualization.



## 4.2 Program description

In this chapter the different programs are explained. The program is structured in folders. Each folder is independent and is described in a separate subchapter.

### 4.2.1 Analog signals

To generate signals an AX522 is used. In the folder AnalogSignals the two POU's are used to generate output signals and read the input signals. If no AX522 module is used, the outputs are used as signal instead of the inputs.

**GenerateSignals** is a program which is called by a separate task. The cycle time of this task is 1 ms. When changing the task time the layout of the analog signals is changing. The eight signals generated in this program are:

1. Ramp
2. Triangle
3. Rectangle
4. Square
5. Sinus
6. Cosinus
7. Random
8. Binary random

The names of the signals are stored in the textlist **SignalNames**.

**ReadInputs** reads the AX522 inputs and saves the variables converted to -10 - +10 V into the global variables list.

The **UnitConversion** is used to convert the analog input variables from an integer value to a real value which is in the range -10 V to + 10 V.



Note: The setting run on config fault is set to true for the IO bus. In case no AX522 is connected, the program can go to run and will automatically use the output values instead of the input values.

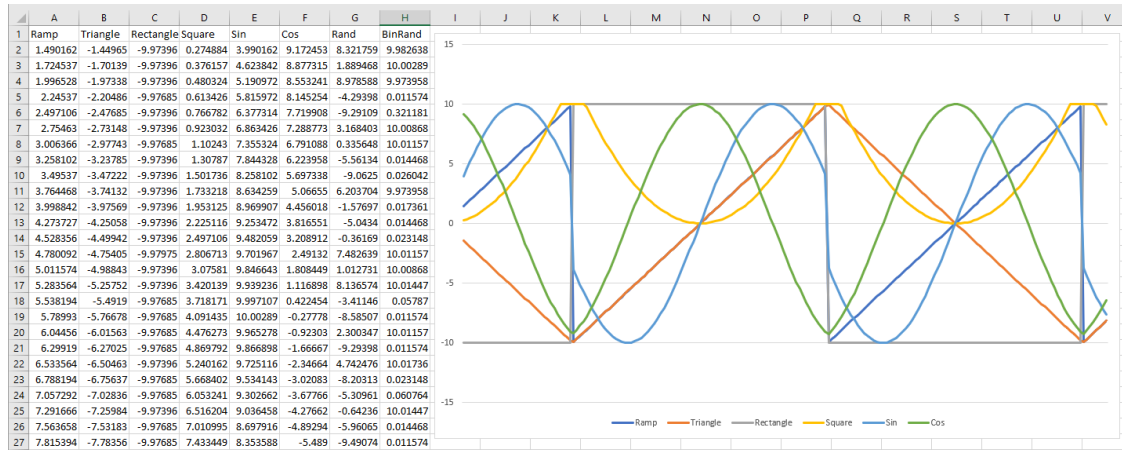
### 4.2.2 Snapshot

In the folder Snapshot the main logic is located. The GVL **SnapshotConst** contains the size of the buffer which is used for logging. This is by default 200 samples and can be changed.

**SaveReadSnapshot** is the program which saves the values into the binary and the text file and reads them back. The two function blocks BinSnapshot (FB) and CsvSnapshot (FB) which are both implementing the interface ITFSnapshot are used to save the data inside a file or read them back. For this, the methods ReadData and SaveData are used.

## 4.3 Usage in computer (Excel)

To get the data on a computer either the SD card, if used, can be plugged to the PC or the file can be transferred via the file browser within Automation Builder. A tool like Excel can be used to display the CSV in lines and columns. The normal tools can be used to plot a line chart with the existing data. The two random numbers are not included to have a better overview. This is displayed in the screenshot below.





## 5 Adding on event

The program Adding on event adds three slider positions to a CSV file on the userdisk. This functionality can be used to log several variables with a timestamp to a CSV file. Then further analysis with this data can be done.

### 5.1 Visualization

On the right side of Main\_VISU the “Adding on event” functionality is located.

By clicking “Add Values” the program will append the existing CSV file with the three slider positions and a timestamp. In case of an error the error number is displayed. Furthermore, the existing CSV can be deleted. If the file has been deleted, a new file is automatically created when saving the values.

On the bottom the content of the CSV is shown. The values are comma separated.

### Adding on event

---

3 Sliders are used.  
Each time the user selects 'Add Values' another line is added to the CSV

---

– Add the slider Position to the CSV –

Program Lines Adding

The current value of the three sliders is added with timestamp to a CSV

Add Values

-100   -75   -50   -25   0   25   50   75   100

-100   -75   -50   -25   0   25   50   75   100

-100   -75   -50   -25   0   25   50   75   100

eErrorId: 0

Delete CSV

---

```

Timestamp;Slider 1;Slider 2;Slider 3
#2023-09-19-09:34:37;-65;87;82
#2023-09-19-09:34:39;-65;-25;82
#2023-09-19-09:34:41;0;-25;12
#2023-09-19-09:34:42;0;35;12
#2023-09-19-09:34:47;-57;35;60
#2023-09-19-09:34:51;-57;-15;90
#2023-09-19-09:34:51;-57;-15;90
#2023-09-19-09:34:56;77;44;18
#2023-09-19-09:34:58;52;44;18
#2023-09-19-09:35:02;8;-67;73
#2023-09-19-09:35:05;8;-67;95
  
```

## 5.2 Program description

This program has the POU *AddingSliderPos* which calls the function block *AddToCsv* (FB).

There is a short description inside the code, a more detailed description is inside this chapter.

**AddToCsv** is the function block which is handling the CSV file. It internally has a step chain which is adding a line to the CSV file whenever the Execute input is triggered.

In the first step (1) it is checked whether the file is existing, or the header must be written.

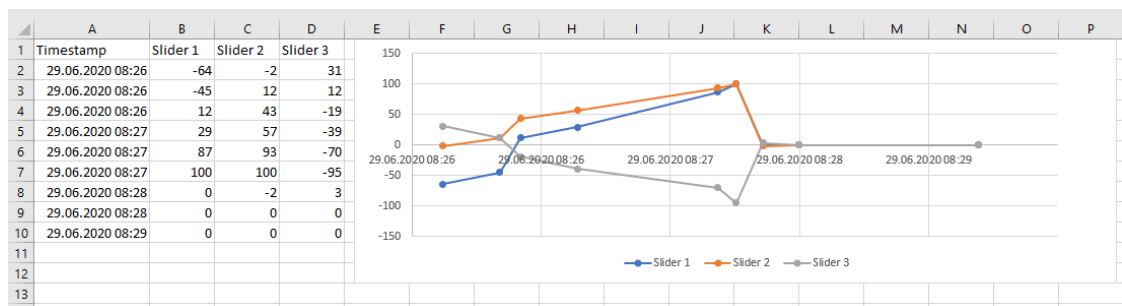
In step 10 the file is opened, or created if not yet existing, and the tree slider positions are added to the file. In step 20 the write to the file buffer is done.

In the step 30 the file is saved and closed afterwards in step 31.

Other functionalities like reading the whole file content or deleting the file are realized in methods.

## 5.3 Usage in computer (Excel)

Similar like described in chapter 4.3 also this file can be analyzed on a computer like visible in the screenshot below.



---

**ABB AG**

Contact:

<https://access.motion.abb.com/contact/contact>

Homepage:

[www.abb.com/plc](http://www.abb.com/plc)

---

We reserve the right to make technical changes or modify the contents of this document without prior notice. With regard to purchase orders, the agreed particulars shall prevail. ABB AG does not accept any responsibility whatsoever for potential errors or possible lack of information in this document.

We reserve all rights in this document and in the subject matter and illustrations contained therein. Any reproduction, disclosure to third parties or utilization of its contents – in whole or in parts – is forbidden without prior written consent of ABB AG.  
Copyright© 2024 ABB. All rights reserved.