|  |  |  |
| --- | --- | --- |
| [FUNCTIONAL SPECIFICATIONS] | | |
| [unCPC recipe component] | | |
| [USER REQUIREMENTS AND FUNCTIONAL SPECIFICATIONS] | | |
| DOCUMENT PREPARED BY: | DOCUMENT CHECKED BY: | DOCUMENT APPROVED BY: |
| G.Thomas  I.Prieto Barreiro | [Checkers] | [Approvers] |

HISTORY OF CHANGES

|  |  |  |  |
| --- | --- | --- | --- |
| REV. NO. | DATE | PAGES | DESCRIPTIONS OF THE CHANGES |
| 1.1  1.2  1.3  1.4 | 2011-05-02  2011-05-04  2011-05-23  2011-06-15 | n | Renamed recipe type to recipe class as notion of type is already used in JCOP component and has functionality.  Functional specifications on activation mechanism added.  Updated recipe class properties. Added new requirements for access control. Added definition of input configuration files.  Recipe activation example added. |

TABLE OF CONTENTS

[1. PVSS User Requirements 4](#_Toc300576286)

[1.1 Functionality 4](#_Toc300576287)

[1.1.1 Recipe Class management 4](#_Toc300576288)

[1.1.2 Recipe management 5](#_Toc300576289)

[2. Functional Activation specifications 7](#_Toc300576290)

[2.1 PLC implementation 7](#_Toc300576291)

[2.1.1 Recipe buffers 7](#_Toc300576292)

[2.1.2 Header and Recipe status buffers 8](#_Toc300576293)

[2.2 PVSS implementation 10](#_Toc300576294)

[2.3 Recipe activation mechanism from PVSS to peripheral (PLC) 13](#_Toc300576295)

[2.3.1 Sequence Diagram 13](#_Toc300576296)

[2.3.2 Assumptions 14](#_Toc300576297)

[2.3.3 Restrictions 14](#_Toc300576298)

[2.4 Communication Tests 14](#_Toc300576299)

[2.4.1 Schneider results 14](#_Toc300576300)

[2.4.2 Siemens results 14](#_Toc300576301)

[2.5 PVSS design specifications 15](#_Toc300576302)

[2.5.1 Access Control configuration 15](#_Toc300576303)

[2.5.2 Import/Export Recipe Configuration file 15](#_Toc300576304)

[2.6 Recipe activation example 16](#_Toc300576305)

[2.6.1 Recipe definition 16](#_Toc300576306)

[2.6.2 Header buffer 17](#_Toc300576307)

[2.6.3 ManReg Address and ManReg Values buffers 18](#_Toc300576308)

[2.6.4 Req Address and Req Values buffers 20](#_Toc300576309)

[2.6.5 End of data transmission 21](#_Toc300576310)

[2.6.6 CRC Algorithm 21](#_Toc300576311)

# PVSS User Requirements

* Concept

Manage/apply/store/reload set of values (set-points, thresholds, commands, etc…) for a set of parameters. Manage multiple set of values of the same set of parameters (several recipes of the same recipe class).

NB: One value corresponds to one target.

## Functionality

### Recipe Class management

#### Definition

A Recipe Class consists of a list of recipe elements (datapoint elements). It represents the parameters (setpoints, thresholds) which belong to a recipe. It does not assign to each parameter a value.

NB: in GCS a recipe class definition in PVSS correspond to a recipe class in PLC with an associated latch.

#### Properties

* RCC\_P1: For a given recipe class, one needs to know:
* Last activated recipe name;
* Last activation time;
* Author of the last activation;
* Status;

#### Access Control

* RCC\_AC1: Need to associate to each recipe class to an Access Control Domain.(to check according to implementation choice)
* RCC\_AC2: Need to associate a recipe to any AC Domain, without any constraints of name or relationship. The choice of the AC Domain cannot be limited to the domains of PCOS.
* RCC\_AC3: Requested privileges for recipe related action should be configurable

There should be a default configuration at the component level and it should be possible to configure the requested privileges per action for each recipe class. The default settings are the following (details in paragraph 2.5.1.1)

* Recipe Class Add/Delete/Edit – Admin;
* Recipe Add/Delete/Edit values – Expert;
* Recipe Activation – Operator;
* Recipe Reading – Monitor;

NB: One Access Control domain should be enough. The associated domain could be inherited from the recipe class. in GCS the domain of a recipe was not the one of the associated PCO.

* RCC\_AC4: Dynamic recipe operations:
* Recipe Class Creation: when creating a new recipe Class one should check first the device access rights. If the user has no rights on the device he should not be able to select it. (only devices for which user has the required privileges should be displayed)
* RCP\_AC1: For the recipe instances operations,
* Recipe instances Activation: No access checking on the access rights of a device for the activation: an operator not allowed to set a value on a device can still activate a recipe including this device (set value on this device).

#### Operations

**Creation (Interactive, batch)**

* RCC\_Cr1: One shall be able to create recipe classes from a configuration files (recipe classes are predefined in PVSS and PLC) [a la GCS]
* RCC\_Cr2: One shall be able to create Recipe classes interactively from panels at run time.

NB1: if created interactively, a solution different than GCS for the activation, address optimization and feedback should be found.

**Edition**

* RCC\_Ed1: one shall be able to view/modify the current available recipe classes and status.
* RCC\_Ed2: One shall be able to modify a recipe class and propagate the changes to the associated recipes. A confirmation message shall be displayed to the user.

**Deletion**

* RCC\_Del1: Deleting a recipe class should automatically delete its associated recipes. A confirmation message shall be displayed to the user.

**Load/Save to DB**

* RCC\_Sav1: One should be able to save a recipe class definition to a database.
* RCC\_Ld1: One should be able to load a recipe class definition from a database.

**Import/Export**

* RCC\_Imp1: One should be able to import a recipe class definition from a file.
* RCC\_Exp1: One should be able to export a recipe class definition to a file.

### Recipe management

#### Definition

A recipe is a collection of values for the recipe elements of the associated recipe classes.

#### Properties

* RCP\_P1: For a given recipe, one needs to know:
* To which recipe class it belongs;
* Last modification time;
* Author of the last modification;
* Last activation time;
* Author of the last activation;
* If it is used in PLC

#### Operations

**Creation (Interactive, batch)**

* RCP\_Cr1: One shall be able to create recipes from a configuration files.
* RCP\_Cr2: One shall be able to create recipes interactively from panels (tables) at run time.

**Edition**

* RCP\_Ed1: One shall be able to edit and modify recipe values interactively from panels (tables) at run time.
* RCP\_Ed2: One shall be able to adopt the online values into an existing recipe or to a new one.

**Deletion**

* RCP\_Del1: One shall be able to delete a recipe from the Recipe Tool editor.

**Load/Save to DB (ORACLE or PVSS local)**

* RCP\_Ld1: One should be able to load a recipe class definition from a database.
* RCP\_Sv1: One should be able to save a recipe class definition to a database.
* RCP\_Cb1: One shall be able to reload a recipe from a database and combine with the online one and keep its persistency. (for upgrade)

**Import/Export from/to file**

* RCP\_IE1: One should be able to import/export a recipes definition to a file.

**Navigation**

* RCP\_NAV1: One shall be able to navigate through the existing recipes easily. The navigation shall be organized according to the PCO hierarchy and to their associated recipe class. The change of context should be helped by pull down lists: List of sub-detector, then updated lists of module (or \*)
* RCP\_NAV2: If PCO have no recipe class associated, they shall not appear in the tree.
* RCP\_NAV3: One shall be able to call the Recipe Editor tool from the PCO menu. The context shall already be selected, if no association top level or level immediately above.

NB: For GCS we were using system and module – not generic enough.

NB: Any users should be able to filter the recipes according to the hierarchy of the plant (i.e. from system to modules and recipe classes).

**Transfer to peripheral (Activation)**

* RCP\_A1: One shall be able to activate large recipes (order of hundreds of values)
* RCP\_A2: One shall have feedback on the activation results.
* RCP\_A3: One shall be able to activate all predefined recipes with initial values from a selected PCO hierarchy.
* RCP\_A4: One shall be able to activate all last activated recipes from a selected PCO hierarchy.(after a PLC crash)
* RCP\_A5: The time required to check the success of the activation of a recipe shall be short (max 30s). This can have an impact on the choice of possible types for recipe elements.
* RCP\_A6: A recipe can only be activated from an operator request.
* RCP\_A6: Only one recipe can be activated at a time. A graphical representation shall be shown to the user to tell him/her if it is already in use. (i.e lock, disable the activation button)

# Functional Activation specifications

## PLC implementation

### Recipe buffers

The PLCs will have several dedicated buffers to store the recipes data coming from PVSS. The size and address of these buffers will be specified in the UAB application parameters. The list of buffers is the following:

* **Header (Word)**: this buffer will contain the header information of the recipe. The header content will be detailed in the following chapters.
* **Recipe Status (Word)**: this buffer will contain the recipe status in the PLC. The content of this buffer will be detailed in the following chapters.
* **ManReg Addresses (Word):** thisbuffer will contain the addresses of the Manual Registers used in the recipe. The maximum size of this buffer is limited to 1000 words.
* **ManReg Values (Word):** this buffer will contain the Manual Register values that have to be written in the addresses specified in the ‘ManReg Addresses’ buffer. The maximum size of this buffer is limited to 1000 words.
* **Request Addresses (Word):** this buffer will contain the addresses of the values used in the recipe. The maximum size of this buffer is limited to 1000 words.
* **Request Values (Float):** this buffer will contain the values that have to be written in the addresses specified in the ‘Word Address values’ buffer. The maximum size of this buffer is limited to 1000 floats.

### Header and Recipe status buffers

The header and status buffers have the same structure for both target platforms (Siemens and Schneider). For the Siemens platform it’s necessary to specify the DB address of each device type, so some buffer positions have only meaning for the Siemens PLCs and are irrelevant for the Schneider PLCs.

The content of the header and recipe status buffers is specified in the following tables :

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Header Buffer** |  |  | **Recipe Status Buffer** |  |
| RecipeId Low | 0 |  | RecipeId Low | 0 |
| RecipeId High | 1 |  | RecipeId High | 1 |
| Nb. ManReg | 2 |  | Nb. ManReg | 2 |
| Nb. of Digital Param. | 3 |  | Nb. of Digital Param. | 3 |
| DB @ of Digital Param. | 4 |  | DB @ of Digital Param. | 4 |
| Nb. of Word Param. | 5 |  | Nb. of Word Param. | 5 |
| DB @ of Word Param. | 6 |  | DB @ of Word Param. | 6 |
| Nb. of Analog Param. | 7 |  | Nb. of Analog Param. | 7 |
| DB @ of Analog Param. | 8 |  | DB @ of Analog Param. | 8 |
| Nb. of Analog Alarm Param. | 9 |  | Nb. of Analog Alarm Param. | 9 |
| DB @ of Analog Alarm Param. | 10 |  | DB @ of Analog Alarm Param. | 10 |
| Nb. of Controller Param. | 11 |  | Nb. of Controller Param. | 11 |
| DB @ of Controller Param. | 12 |  | DB @ of Controller Param. | 12 |
| Sent value | 13 |  | Recipe Status | 13 |
| CRC | 14 |  | CRC Status | 14 |
| *<Empty>* | 15 |  | Error Detail | 15 |
| *<Empty>* | 16 |  | *<Empty>* | 16 |
| *<Empty>* | 17 |  | *<Empty>* | 17 |
| *<Empty>* | 18 |  | *<Empty>* | 18 |
| *<Empty>* | 19 |  | *<Empty>* | 19 |

Where :

* **RecipeId Low** : recipe unique identifier (low part). The unique ID number generated from PVSS is a 32 bit integer. To transmit this value to the PLC, the 32 bit integer is split in two parts and transmitted as two 16 bit integers.
* **RecipeId High** : recipe unique identifier (high part).
* **Nb. of ManReg** : number of Manual Registers used in the recipe.
* **Nb. of Digital Param.** : number of Digital Parameters used in the recipe.
* **DB @ of Digital Param.** **(Siemens only):** is the DB number where the Digital Parameters are located.
* **Nb. of Word Param.** : number of Word Parameters used in the recipe.
* **DB @ of Word Param.** **(Siemens only)**: is the DB number where the Word Parameters are located.
* **Nb. of Analog Param.** : number of Analog Parameters used in the recipe.
* **DB @ of Analog Param.** **(Siemens only)**: is the DB number where the Analog Parameters are located.
* **Nb. of Analog Alarm Param.** : number of Analog Alarm Parameters used in the recipe.
* **DB @ of Analog Alarm Param.** **(Siemens only)**: is the DB number where the Analog Alarm Parameters are located.
* **Nb. of Controller Param.** : number of Controller Parameters used in the recipe.
* **DB @ of Controller Param.** **(Siemens only)**: is the DB number where the Controller Parameters are located.
* **Sent value**: this values is used to indicate that all the recipe values has been sent to the PLC. The possible values are:
  + 1: New recipe activation is started.
  + 2: Recipe values sent. All the data has been sent to the PLC recipe buffers.
  + 3: Activate recipe. The PLC must activate the recipe with the data stored in the recipe buffers.
* **CRC**: Cyclic redundancy check code computed by PVSS.
* **CRC Status**: Cyclic redundancy check code status. The possible values are:
  + -1: Bad CRC
  + 0: Standby
  + 1: Ok
* **Error Detail**: 0=unknown, 1=Time Out PLC, 2=CRC problem,…

## PVSS implementation

Due to a limitation in the arrays size of the MODBUS protocol (240 bytes), the data point types used to define the recipe buffers can’t contain integer arrays bigger than 120 elements or float arrays bigger than 60 elements.

The structure of the data point type used to define the recipe buffers in PVSS is the following :

* **\_Recipe\_Buffers\_<PLC\_Name>** [Structure]
  + **Header** [Int Structure]
    - RecipeIdLow
    - RecipeIdHigh
    - NumManReg
    - NumDigitalPar
    - DbDigitalPar
    - NumWordPar
    - DbWordPar
    - NumAnalogPar
    - DbAnalogPar
    - NumAAlarmPar
    - DbAAlarmPar
    - NumPID
    - DbPID
    - SentValue
    - Crc
    - Value16
    - Value17
    - Value18
    - Value19
    - Value20
  + **Recipe Status** [Int Structure]
    - RecipeIdLow
    - RecipeIdHigh
    - NumManReg
    - NumDigitalPar
    - DbDigitalPar
    - NumWordPar
    - DbWordPar
    - NumAnalogPar
    - DbAnalogPar
    - NumAAlarmPar
    - DbAAlarmPar
    - NumControllerPar
    - DbControllerPar
    - RecipeStatus
    - crcStatus
    - ErrorDetail
    - Value17
    - Value18
    - Value19
    - Value20
  + **ManRegAddr** [Structure]
    - Structure1 [Int Structure]
      * Value1
      * Value2
      * …
      * Value120
    - Structure2 [Int Structure]
      * Value121
      * Value122
      * …
      * Value240

...

* + - Structure9[Int Structure]
      * Value961
      * …
      * Value1000
  + **ManRegVal** [Structure]
    - Structure1 [Int Structure]
      * Value1
      * Value2
      * …
      * Value120
    - Structure2 [Int Structure]
      * Value121
      * Value122
      * …
      * Value240

...

* + - Structure9 [Int Structure]
      * Value961
      * …
      * Value1000
  + **ReqAddr** [Structure]
    - Structure1 [Int Structure]
      * Value1
      * Value2
      * …
      * Value120
    - Structure2 [Int Structure]
      * Value121
      * Value122
      * …
      * Value240

...

* + - Structure9 [Int Structure]
      * Value961
      * …
      * Value1000
  + **ReqVal** [Structure]
    - Structure1 [Float Structure]
      * Value1
      * Value2
      * …
      * Value60
    - Structure2 [Float Structure]
      * Value61
      * Value62
      * …
      * Value120

...

* + - Structure17 [Float Structure]
      * Value961
      * …
      * Value1000

## Recipe activation mechanism from PVSS to peripheral (PLC)

### Sequence Diagram



Figure - Sequence diagram for the recipe activation mechanism.

### Assumptions

* The address and size of the PLC buffers is predefined in the UAB application parameters.
* The PLC is ‘locked’ while it’s receiving a recipe (it’s not possible to initiate the transfer of a different recipe).

### Restrictions

* It’s only possible to send a unique recipe at a certain moment. Even if there are several users logged in one application at the same time, it’s only possible to send one recipe simultaneously.

## Communication Tests

The communication tests performed to check the feasibility to transfer large blocks of data are the following (each transfer test was done using a unique call to the dpSetWait() method in PVSS):

1. Transfer of 1000 integers from PVSS to the PLC.
2. Transfer of 1000 floats from PVSS to the PLC.
3. Transfer of 3000 integers and 1000 floats from PVSS to the PLC.

The results of each test contain the number of TCP packages needed to transfer all the data and the average transmission time used to send all the packages.

### Schneider results

|  |  |  |
| --- | --- | --- |
| **Test ID** | **Nb. of packages (TCP)** | **Avg. transmission time (sec.)** |
| 1 | 9 | 0.26 |
| 2 | 17 | 0.45 |
| 3 | 44 | 1.37 |

### Siemens results

|  |  |  |
| --- | --- | --- |
| **Test ID** | **Nb. of packages (TCP)** | **Avg. transmission time (sec.)** |
| 1 | 10 | 4.28 |
| 2 | 20 | 8.85 |
| 3 | 49 | 23.24 |

## PVSS design specifications

### Access Control configuration

#### Default Privileges/Actions

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Action/privileges** | **Monitor** | | **Operator** | | **Expert** | | **Administrator** | | **Comments** | |
| Recipes Operations | | | | | | | | | | |
| Activate recipes |  | | X | |  | |  | |  | |
| Adopt online values for recipe |  | |  | | X | |  | |  | |
| Group Activation (of initial recipes) |  | |  | |  | |  | |  | |
| Group of Activation (Last Activated Recipes) |  | |  | |  | |  | |  | |
| Duplicate recipe |  | |  | | X | |  | |  | |
| Create recipe |  | |  | | X | |  | |  | |
| Delete recipe |  | |  | | X | |  | |  | |
| Modify recipe |  | |  | | X | |  | |  | |
| Save recipe |  | |  | | X | |  | |  | |
|  |  | |  | |  | |  | |  | |
| Recipe Class Operations | | | | | | | | | | |
| Create | |  | |  | |  | | X | |  |
| Delete | |  | |  | |  | | X | |  |
| Modify | |  | |  | |  | | X | |  |
| Save | |  | |  | |  | | X | |  |
|  | |  | |  | |  | |  | |  |

#### Implementation

### Import/Export Recipe Configuration file

#### Recipe Class/Recipe instances required configuration settings

* Declaration of recipe Classs and recipe instances should be separated.
* Id number of recipe instance
* It is possible to have any number of instances for a given Class.
* An Access Control Domain, requested privileges per action shall be specified at the level of Class declaration.
* In the recipe Class declaration only system + device alias name shall be specified; Q: are indexes required to sort elements ?.
* Predefined Recipe Classes and instances (“initial” or “custom”) is specified in an instance name (“Initial” -suffix).
* PCO hierarchy navigation settings

## Recipe activation example

To clarify the activation mechanism procedure, we will see a step by step example for Siemens. The same mechanism will be used in the Schneider PLCs, but in this case the DB address data in the recipe header buffer will always contain the value ‘0’. The recipe used in the example is made of the following UNICOS devices:

* 2 Digital parameters
* 2 Word parameters
* 2 Analog parameters
* 1 Analog alarm
* 1 Controller

### Recipe definition

The following table contains the meaningful information of the devices used in the example :

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Device type** | **Device alias** | **DPE name** | **Siemens DPE address** | **Schneider DPE address** | **Recipe Value** |
| DigitalParameter | DPar1 | ManReg01 | DB203.DBW0 | 0 | TRUE |
| DigitalParameter | DPar2 | ManReg01 | DB203.DBW2 | 2 | FALSE |
| WordParameter | WPar1 | ManReg01 | DB213.DBW4 | 4 | - |
| MPosR | DB213.DBW6 | 6 | 100 |
| WordParameter | WPar2 | ManReg01 | DB213.DBW8 | 8 | - |
| MPosR | DB213.DBW10 | 10 | 200 |
| AnalogParameter | APar1 | ManReg01 | DB223.DBW12 | 12 | - |
| MPosR | DB223.DBD14F | 14 | 300.5 |
| AnalogParameter | APar2 | ManReg01 | DB223.DBW18 | 18 | - |
| MPosR | DB223.DBD20F | 20 | 400.5 |
| AnalogAlarm | AAlarm1 | ManReg01 | DB311.DBW24 | 24 | - |
| HH | DB311.DBD28F | 28 | 1000.5 |
| H | DB311.DBD32F | 32 | 900.5 |
| L | DB311.DBD36F | 36 | 200.5 |
| LL | DB311.DBD40F | 40 | 100.5 |
| Controller | Controller1 | ManReg01 | DB292.DBW44 | 44 | - |
| MSP | DB292.DBD48F | 48 | 90.625 |
| MSPH | DB292.DBD52F | 52 | 95.0 |
| MSPL | DB292.DBD56F | 56 | 85.0 |
| MOutH | DB292.DBD60F | 60 | 15.0 |
| MOutL | DB292.DBD64F | 64 | 5.0 |
| MKc | DB292.DBD68F | 68 | 1.0 |
| MTi | DB292.DBD72F | 72 | 2.0 |
| MTd | DB292.DBD76F | 76 | 3.0 |
| MTds | DB292.DBD80F | 80 | 4.0 |

### Header buffer

The values of the Header buffer are filled according to the buffer definition. The buffer positions in gray background only make sense for Siemens, for the Schneider recipes the content of these positions will be always 0.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Header Buffer Definition** |  |  | **Header Buffer Content** |  |
| RecipeId Low | 0 |  | 52107 | 0 |
| RecipeId High | 1 |  | 19960 | 1 |
| Nb. ManReg | 2 |  | 19 | 2 |
| Nb. of Digital Param. | 3 |  | 2 | 3 |
| DB @ of Digital Param. | 4 |  | 203 | 4 |
| Nb. of Word Param. | 5 |  | 2 | 5 |
| DB @ of Word Param. | 6 |  | 213 | 6 |
| Nb. of Analog Param. | 7 |  | 2 | 7 |
| DB @ of Analog Param. | 8 |  | 223 | 8 |
| Nb. of Analog Alarm Param. | 9 |  | 4 | 9 |
| DB @ of Analog Alarm Param. | 10 |  | 311 | 10 |
| Nb. of PID Param. | 11 |  | 9 | 11 |
| DB @ of PID Param. | 12 |  | 292 | 12 |
| Sent value | 13 |  | 0 | 13 |
| CRC | 14 |  | \* To be specified | 14 |

### ManReg Address and ManReg Values buffers

These buffers will contain the ManReg addresses and values of the recipe devices. For each XParameter device in the recipe it will be one ManReg address and one ManReg value in the buffers. For the AnalogAlarm and Controller devices the number of items in the buffers depends on the number of the device parameters included in the recipe (HH, H, L, LL, …).

The ManReg Address buffer contains the ManReg01 addresses of the recipe devices in the following order :

1. DigitalParameter devices
2. WordParameter devices
3. AnalogParameter devices
4. AnalogAlarm devices
5. Controller devices

**Note** : in the case of Siemens, the value in the ManReg Address buffer is the offset in the corresponding DB where the register is located. In the case of Scheider, the ManReg Address buffer contains the full address where the register is located.

The ManReg Values buffer contains the values to write in the ManReg registers. The values of this buffer are computed as follows :

1. DigitalParameter
   * Bit 2 (ArmRcp) : contains the value ‘1’
   * Bit 4 (MOnR) : contains the ‘recipe value’ of the Digital Parameter. In the example, it will be ‘1’ for the DPar1 and ‘0’ for the DPar2.
   * Bit 5 (MOffR) : contains the negation of the ‘recipe value’ of the Digital Parameter. In the example, i twill be ‘0’ for the DPar1 and ‘1’ for the DPar2.
2. WordParameter
   * Bit 2 (ArmRcp) : contains the value ‘1’
   * Bit 6 (MNewMR) : contains the value ‘1’
   * The ManReg01 value in decimal is 68
3. AnalogParameter
   * Bit 2 (ArmRcp) : contains the value ‘1’
   * Bit 6 (MNewMR) : containes the value ‘1’
   * The ManReg01 value in decimal is 68
4. AnalogAlarm
   * Bit 2 (ArmRcp) : contains the value ‘1’
   * Bit 8 (MNewHHR) : contains the value ‘1’
   * Bit 9 (MNewHR) : contains the value ‘1’
   * Bit 12 (MNewLR) : contains the value ‘1’
   * Bit 13 (MNewLLR) : contains the value ‘1’
   * The ManReg01 value in decimal is 13060
5. Controller
   * Bit 4 (ArmRcp) : contains the value ‘1’
   * Bit 7 (MNewSPR) : contains the value ‘1’
   * Bit 8 (MNewSPHLiR) : contains the value ‘1’
   * Bit 9 (MNewSPLLiR) : contains the value ‘1’
   * Bit 10 (MNewOutHLiR) : contains the value ‘1’
   * Bit 11 (MNewOutLLiR) : contains the value ‘1’
   * Bit 12 (MNewKcR) : contains the value ‘1’
   * Bit 13 (MNewTdR) : contains the value ‘1’
   * Bit 14 (MNewTiR) : contains the value ‘1’
   * Bit 15 (MNewTdsR) : contains the value ‘1’
   * The ManReg01 value in decimal is 65424

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **ManReg Address Buffer** |  |  | **ManReg Values Buffer** |  |
| DPar1 | 0 | 0 |  | 20 | 0 |
| DPar2 | 2 | 1 |  | 36 | 1 |
| WPar1 | 4 | 2 |  | 68 | 2 |
| WPar2 | 8 | 3 |  | 68 | 3 |
| APar1 | 12 | 4 |  | 68 | 4 |
| APar2 | 18 | 5 |  | 68 | 5 |
| AAlarm1 | 24 | 6 |  | 13060 | 6 |
|  | 24 | 7 |  | 13060 | 7 |
|  | 24 | 8 |  | 13060 | 8 |
|  | 24 | 9 |  | 13060 | 9 |
| Controller1 | 44 | 10 |  | 65424 | 10 |
|  | 44 | 11 |  | 65424 | 11 |
|  | 44 | 12 |  | 65424 | 12 |
|  | 44 | 13 |  | 65424 | 13 |
|  | 44 | 14 |  | 65424 | 14 |
|  | 44 | 15 |  | 65424 | 15 |
|  | 44 | 16 |  | 65424 | 16 |
|  | 44 | 17 |  | 65424 | 17 |
|  | 44 | 18 |  | 65424 | 18 |

### Req Address and Req Values buffers

The Req Address buffer contains the addresses where the recipe values have to be written in the following order :

1. WordParameter devices
2. AnalogParameter devices
3. AnalogAlarm devices
4. Controller devices

**Note** : in the case of Siemens, the value in the Req Address buffer is the offset in the corresponding DB where the value has to be written. In the case of Scheider, the Req Address buffer contains the full address where the value has to be written.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Req Address Buffer** |  |  | **Req Values Buffer** |  |
| WPar1.MPosR | 6 | 0 |  | 100.0 | 0 |
| WPar2.MPosR | 10 | 1 |  | 200.0 | 1 |
| APar1.MPosR | 14 | 2 |  | 300.5 | 2 |
| APar2.MPosR | 20 | 3 |  | 400.5 | 3 |
| AAlarm1.HH | 28 | 4 |  | 1000.5 | 4 |
| AAlarm1.H | 32 | 5 |  | 900.5 | 5 |
| AAlarm1.L | 36 | 6 |  | 200.5 | 6 |
| AAlarm1.LL | 40 | 7 |  | 100.5 | 7 |
| Controller1.MSP | 46 | 8 |  | 90.625 | 8 |
| Controller1.MSPH | 50 | 9 |  | 95.0 | 9 |
| Controller1.MSPL | 54 | 10 |  | 85.0 | 10 |
| Controller1.MOutH | 58 | 11 |  | 15.0 | 11 |
| Controller1.MOutL | 62 | 12 |  | 5.0 | 12 |
| Controller1.MKc | 66 | 13 |  | 1.0 | 13 |
| Controller1.MTi | 70 | 14 |  | 2.0 | 14 |
| Controller1.MTd | 74 | 15 |  | 3.0 | 15 |
| Controller1.MTds | 78 | 16 |  | 4.0 | 16 |

### End of data transmission

* + - When all the recipe data has been sent to the PLC, PVSS will set the ‘Sent value’ of the recipe header buffer (position 13) to ‘2’.
    - Then the PLC will check the received data, calculate the CRC value and write the recipe status and CRC status in the recipe status buffer (positions 13 and 14). The possible values for the recipe status are :
      * -1 : Error
      * 0 : Stand by
      * 1 : New recipe identifier received
      * 2 : Checking CRC
      * 3 : CRC ok, waiting for Active
      * 4 : Active received, executing recipe
      * 5 : Recipe done

And the possible values for the CRC status are :

* + -1 : Bad CRC
  + 1 : Ok
    - If the data transmission was successful, PVSS will the ‘Sent value’ to ‘3’ in the recipe header buffer.

### CRC Algorithm

The CRC algorithm will take the following data as input :

* Header buffer values, positions from 0 to 12 inclusive
* ManReg Address buffer values
* ManReg Values buffer values
* Req Address buffer values
* Req Values buffer values

The result of the algorithm is a 16 bit value that will be sent in the header buffer (position 14). To compute the CRC value, a XOR function of all the input data will be used. The float values (32 bits) will be splitted in two parts before applying the XOR function. The complete procedure to calculate the CRC in our example is the following :