User Manual

ACASYA for WFC

Automatic Control and Analysis System Application for Wheel Fitted Component

September 10, 2020

Continental Automative SAS
1 Avenue Paul Ourliac
BP 83649
31036 TOULOUSE CEDEX 1

TABLE OF CONTENTS

Chapter 1

ACASYA – FIRST STEPS

I. ACASYA – FIRST STEPS

ACASYA for WFC is the brand-new version of the software formerly called Wheel Unit Testing Tool 2.0. This new version has a brand-new design and new improved features. ACASYA allows you to automate the control and analysis of the LSE test bench. In other words, you have the possibility with this application to create your own Test Case scripts, execute them and get a synthesis of the test analysis.

A. Installation

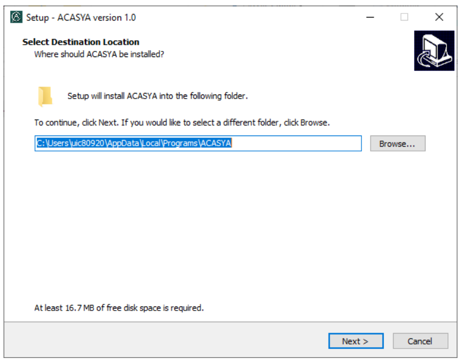
Minimum system requirements:

- OS: Windows 10
- Architecture: 32 or 64-Bit
- CPU: Dual-core CPU
- Memory: 4 GB RAM
- HD free space: 2 GB
- Screen resolution: 1280x1024 pixels display
- Software: Microsoft Office Excel – v. 1908 (build 11929.20934)

Installation procedure:

Double-clicking on ACASYA-setup.exe will launch the installer. Follow then the instructions.

Step 1: Choose where to install ACASYA on your hard drive. The installation folder will contain all the necessary files for the proper using of ACASYA. See the list of installation files in the appendix.


Setup - ACASYA version 1.0

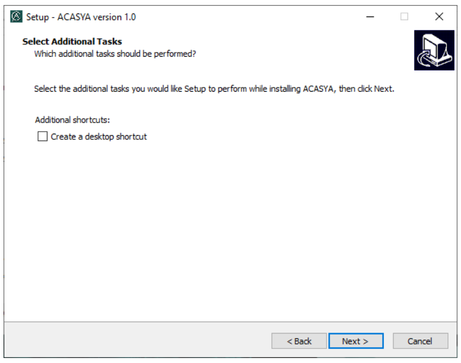
Select Destination Location
Where should ACASYA be installed?

Setup will install ACASYA into the following folder.

To continue, click Next. If you would like to select a different folder, click Browse.

C:\Users\uic80920\AppData\Local\Programs\ACASYA

At least 16.7 MB of free disk space is required.


Setup - ACASYA version 1.0

Select Additional Tasks

Which additional tasks should be performed?

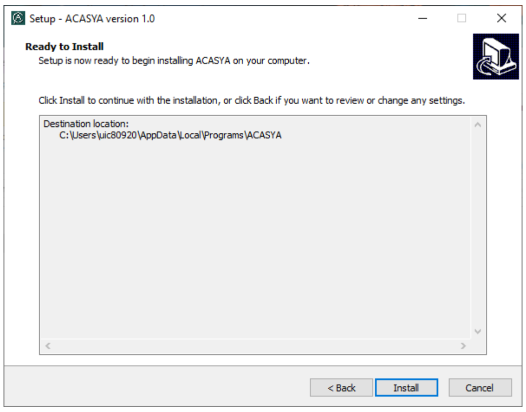
Select the additional tasks you would like Setup to perform while installing ACASYA, then click Next.

Additional shortcuts:

☐ Create a desktop shortcut

Step 2: The installer is able to add a shortcut on the desktop. Check the option if you need it.

Step 3: ACASYA is now ready to be installed. Click on **Install** to launch the installation. The operation might take few seconds.

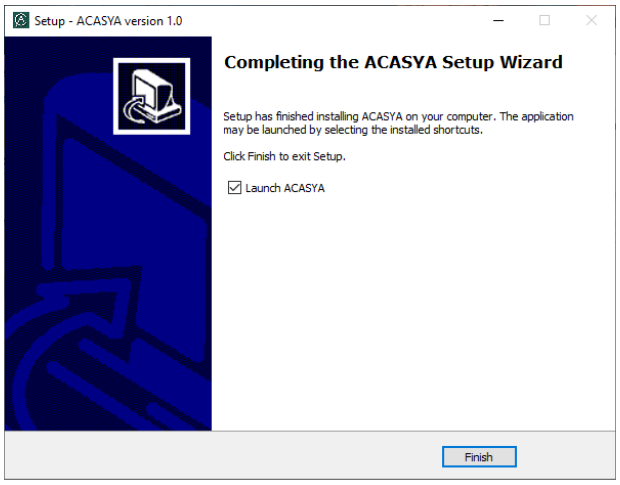

Setup - ACASYA version 1.0

Ready to Install

Setup is now ready to begin installing ACASYA on your computer.

Click Install to continue with the installation, or click Back if you want to review or change any settings.

Destination location:
C:\Users\uic80920\AppData\Local\Programs\ACASYA


Setup - ACASYA version 1.0

Completing the ACASYA Setup Wizard

Setup has finished installing ACASYA on your computer. The application may be launched by selecting the installed shortcuts.

Click Finish to exit Setup.

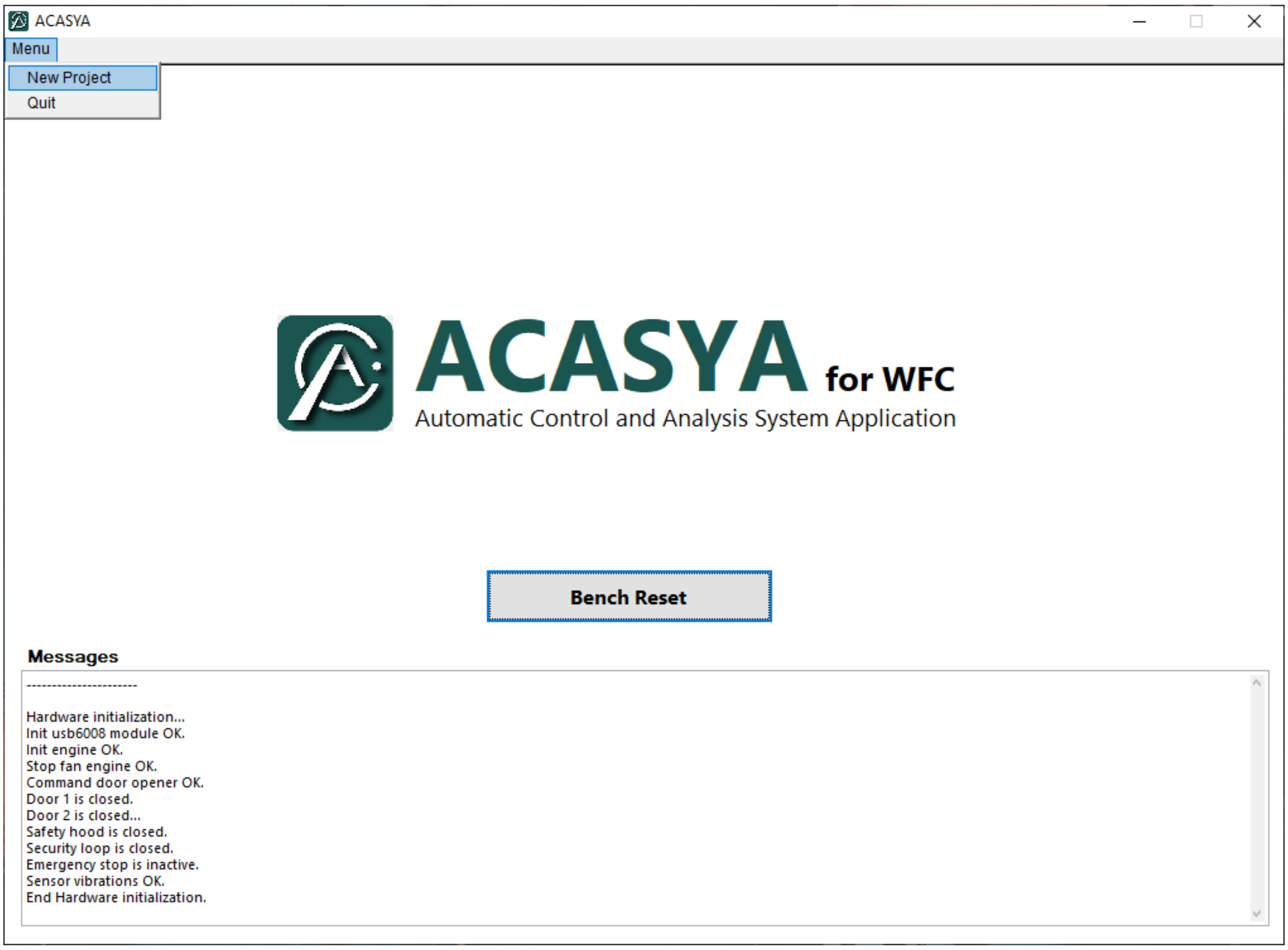
☑ Launch ACASYA

Step 4: The installer informs you when the installation is complete. Check **Launch ACASYA** to immediately launch ACASYA when you click on Finish. Otherwise, click on the ACASYA icon in the Start menu or on the Desktop.

Congratulations, you have now installed ACASYA!

B. Welcome screen

Once ACASYA is launched, the Welcome screen is displayed, as shown in the Figure 1. This Welcome screen indicates the status of the LSE Test Bench and allows you to create a new project.



```
----------------------

Hardware initialization...
Init usb6008 module OK.
Init engine OK.
Stop fan engine OK.
Command door opener OK.
Door 1 is closed.
Door 2 is closed...
Safety hood is closed.
Security loop is closed.
Emergency stop is inactive.
Sensor vibrations OK.
End Hardware initialization.
```

Figure 1. Welcome screen

To create a new project, click on the ***Menu*** tab, then click on ***New Project***.

C. Main screen

After clicking on ***New Project***, the Main screen of ACASYA is displayed, as shown in the Figure 2. This window is organized as follows:

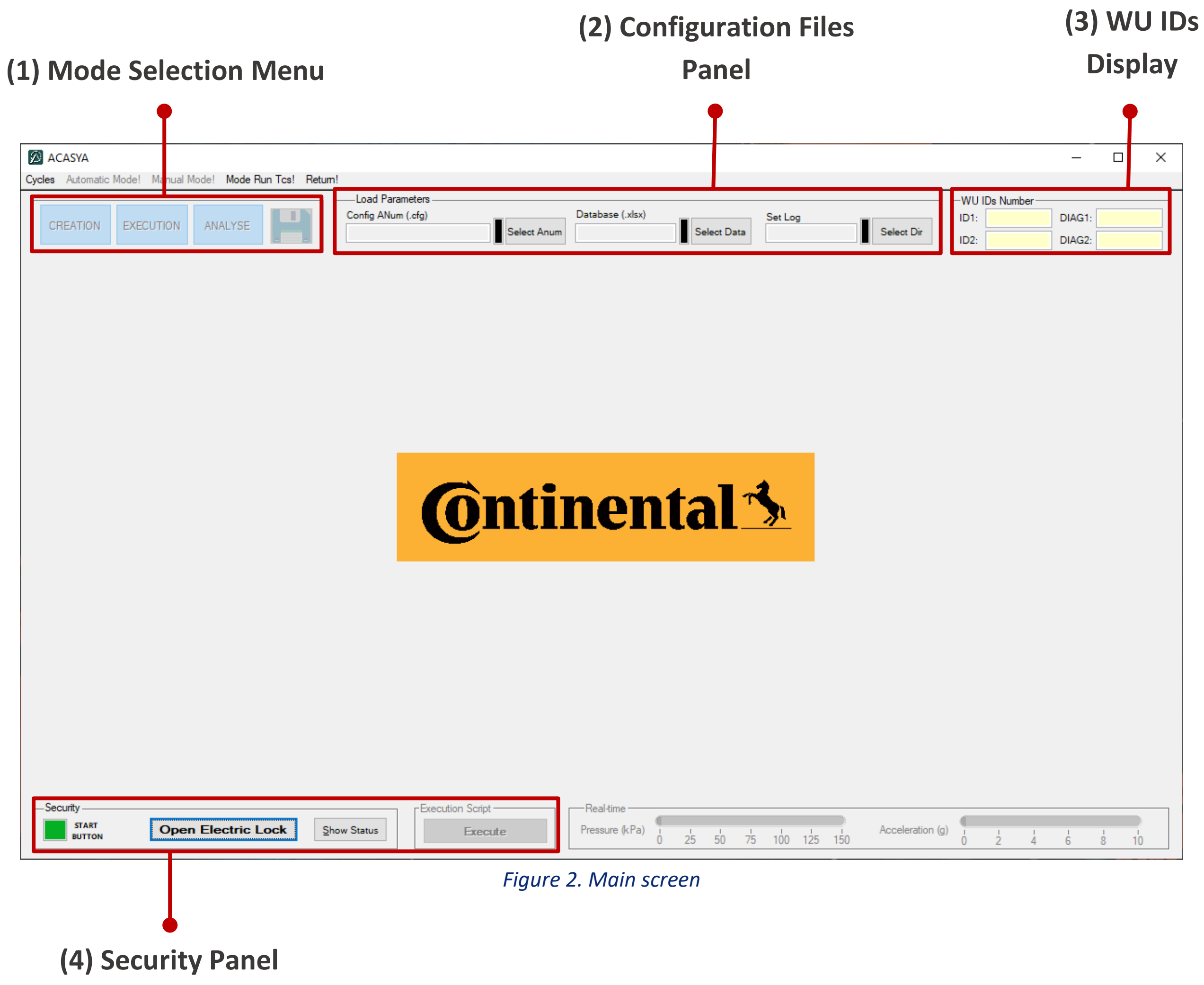


Figure 2. Main screen

(1) Mode Selection Menu: gives access to the 3 modes (Creation, Execution and Analyse).

(2) Configuration Files Panel: allows to load the configurations files and working folder.

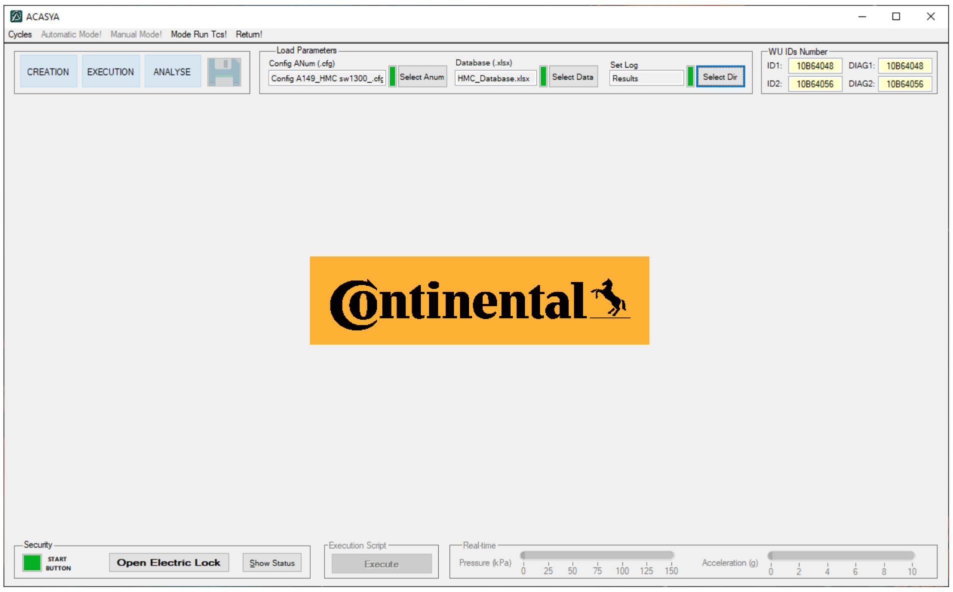
(3) WU IDs Display: shows the WU ID numbers of the WFC defined in the Database.

(4) Security Panel: indicates security status and gives access to some security tool presented in the Security and monitoring chapter.

• Before creating or executing Test Case scripts, ACASYA must be configured imperatively. The user must load the configuration files and the working folder in the Configuration Files Panel presented above. There are three different elements to load[1]:

- ANum configuration file (.cfg file)
- Database configuration file (.xlsx file) - Project specifics parameters file
- Storage folder for logs - a directory which will be stocked the results of the tests.

• Once this step is completed, the ***Creation***, ***Execution*** and ***Analyse*** buttons of the Mode Selection Menu are enabled and the WU IDs from the Database are loaded into the WU IDs Display, as presented in the Figure 3. Main window after configuration.



Load Parameters
Config ANum (.cfg): Config A149_HMC sw1300_.cfg
Database (.xlsx): HMC_Database.xlsx
Set Log: Results

WU IDs Number
ID1: 10B64048 DIAG1: 10B64048
ID2: 10B64056 DIAG2: 10B64056

Security: START BUTTON
Execution Script
Pressure (kPa)
Acceleration (g)

Figure 3. Main window after configuration

[1] The loading order of files and folder is not important.

Chapter 2

CREATION MODE

II. CREATION MODE

In this mode you can create the Test Case, including the Test script and the Expected Results. The main window is splited in two parts: the one on the left side is dedicated to the creation of the Test script and the one on the right side is dedicated to the creation of the Expected Results.

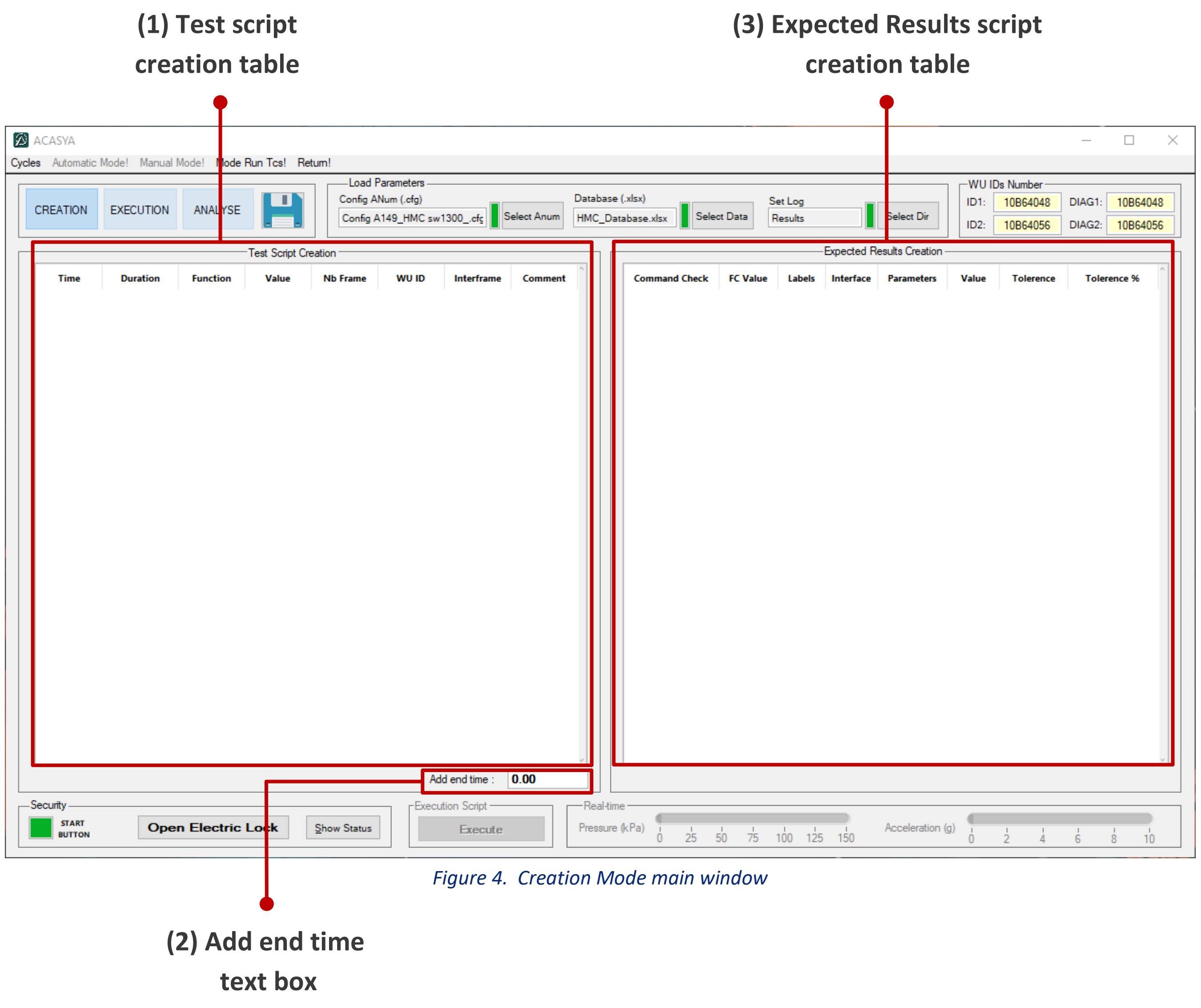


Figure 4. Creation Mode main window

(1) Test script creation table: table to create the Test script (SetA, SetP, SendLFD…).

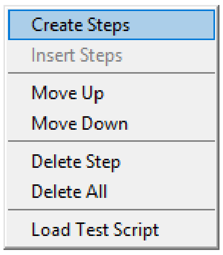
(2) Expected Results script creation table: table to create the Expected Results script (with check functions for the analysis).

(3) Add end time text box: text box to add some extra time to your Test script.

A. Test script creation

Located at the left side of the window, this panel allows you to create the Test script. The Test script contains all the commands (set acceleration, set pressure, send LFD…) that will be executed by the bench during the test.

1. Create a new command



A right click on the table gives you access to several features.
Click on ***Creates Steps*** to add a step to your table.

A new window titled "Script definition" appears, as presented above in the Figure 5.

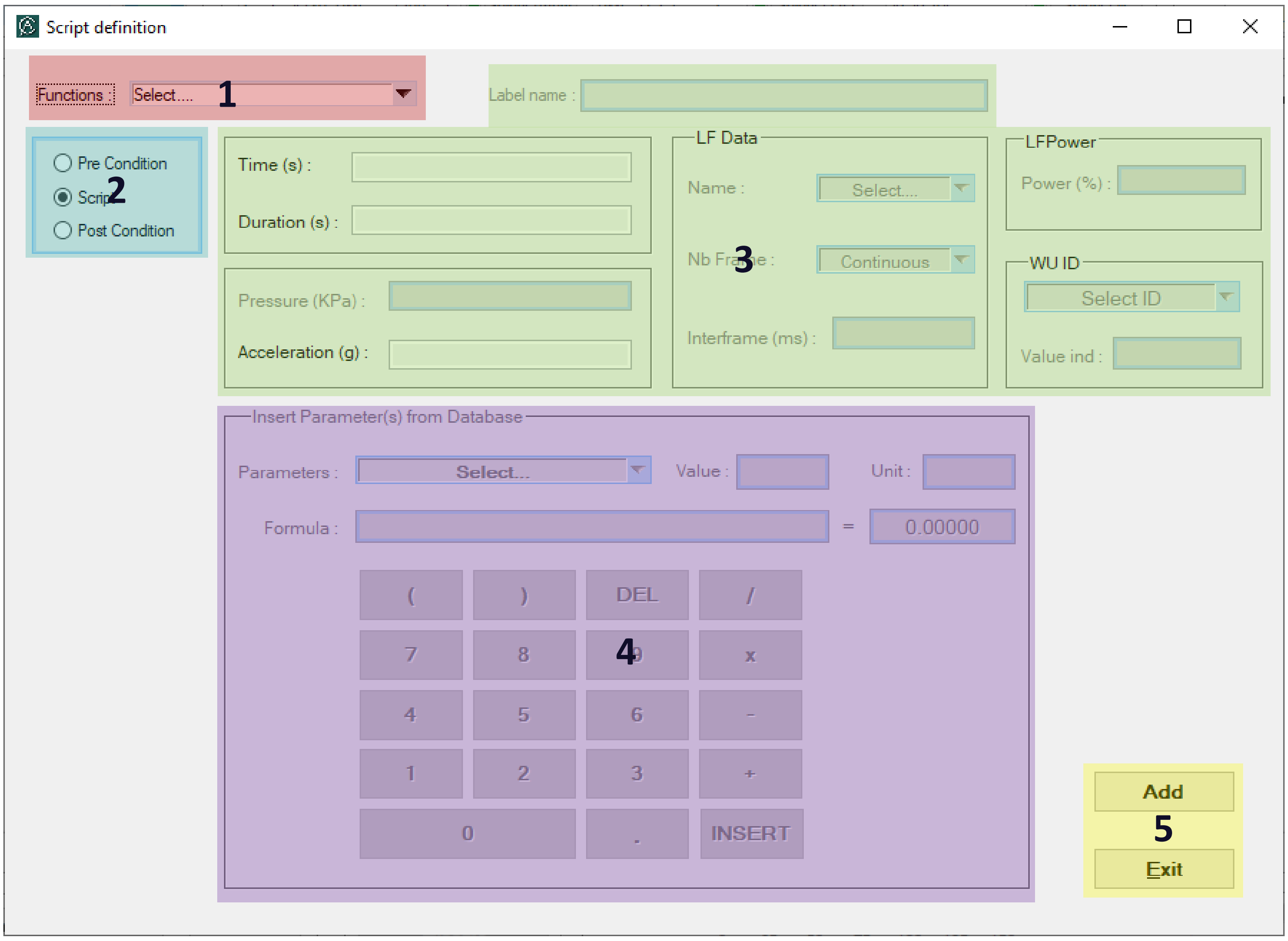


Figure 5. Script definition window

The window is divided as follow:

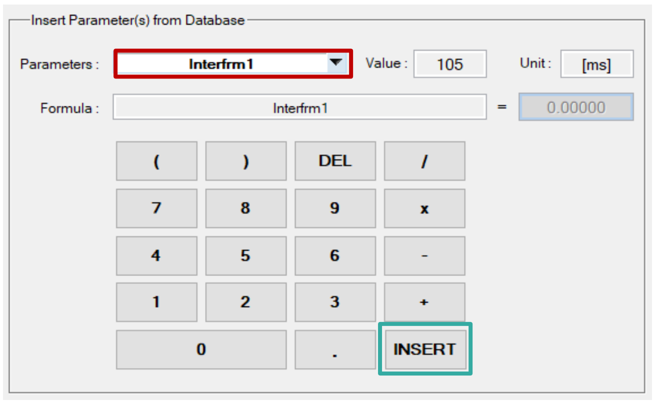
(1) **Function selection:** this contextual menu allows you to choose the command to be executed: (SetA, SetP, SendLFD, StopLFD, Label…). The various commands are detailed in the *Commands details* section.

(2) **Type of command:** this panel allows you to select the type of command. Three types are defined:

- ***Pre condition*****:** command to be executed before the test.
- ***Script*****:** command to be executed during the test.
- ***Post condition*****:** command to be executed after the test.

(3) **Parameters edition:** this panel allows you to configure all parameters linked with the command selected. The parameters are dimmed or not, according to the selected command. To enter a new parameter value, you must use the numeric keypad, presented in section 4.

(4) **Numeric keypad:** the numeric keypad, presented below in the Figure 6, allows you to enter the value of a parameter (Time, Duration, Pressure…).


Insert Parameter(s) from Database

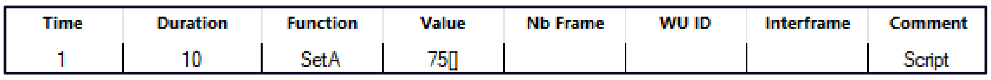
Parameters : Interfrm1 Value : 105 Unit : [ms]

Formula : Interfrm1 = 0.00000

Figure 6. Numeric keypad for Test script Creation

The value can be a numeric value (e.g. "102", "10") or a formula using parameters from the Database. You can choose a specific parameter from the Database clicking on the contextual menu titled "**Parameters**" The unit and the value are displayed next to the parameter's name. To validate the parameter value, click on the **INSERT** button.

(5) **Add and Exit buttons:** once the command has been detailed, you can add it to the table using the button **Add**. A new line like this one is added to the table:



Time	Duration	Function	Value	Nb Frame	WU ID	Interframe	Comment
1	10	SetA	75▯				Script

You can quit the "Script definition" window at any time with the button **Exit**.

2. Commands details

As we have seen above, the Script definition window allows you to configure various commands. The Figures below detail some important information about commands.

a. SetA

Command	**SetA**
Interface	Bench Test
Mandatory parameters	Time [s] Duration [s] Acceleration value [g]
Description	Sets the test bench acceleration to a given value. Starts at time ***Time*** and produces a ramp from an initial acceleration value to the ***Acceleration value*** during ***Duration*** seconds.

b. SetP

Command	**SetP**
Interface	Bench Test
Mandatory parameters	Time [s] Duration [s] Pressure value [kPa]
Description	Sets the test bench pressure to a given value. Starts at time ***Time*** and produces a ramp from an initial pressure value to the ***Pressure value*** during ***Duration*** seconds.

c. SendLFD

Command	**SendLFD**
Interface	ANum
Mandatory parameters	Time [s] LF Data (Name, Nb Frame, Interframe) LFPower [%] WU ID
Description	Sends one or several ***LF Data*** frames to specific ***WU ID*****(s)** with a signal strength at ***LFPower*** %. Starts at time ***Time*** and must be stopped with **StopLFD** command.

d. StopLFD

Command	**StopLFD**
Interface	ANum
Mandatory parameters	Time [s]
Description	Stops sending all LF Data frames at time ***Time***.

e. SendLFCw

Command	**SendLFCw**
Interface	ANum
Mandatory parameters	Time [s] LF Data (Name, Nb Frame, Interframe) LFPower [%] WU ID
Description	Sends one or several ***LF Carrier wave*** frames to specific ***WU ID*****(s)** with a signal strength at ***LFPower*** %. Starts at time ***Time*** and must be stopped with **StopLFCw** command.

f. StopLFCw

Command	**StopLFCw**
Interface	ANum
Mandatory parameters	Time [s]
Description	Stops sending all LF Carrier wave frames at time ***Time***.

g. LFPower

Command	**LFPower**
Interface	ANum
Mandatory parameters	Time [s] LFPower [%]
Description	Sets the signal strength at ***LFPower*** % at time ***Time***.

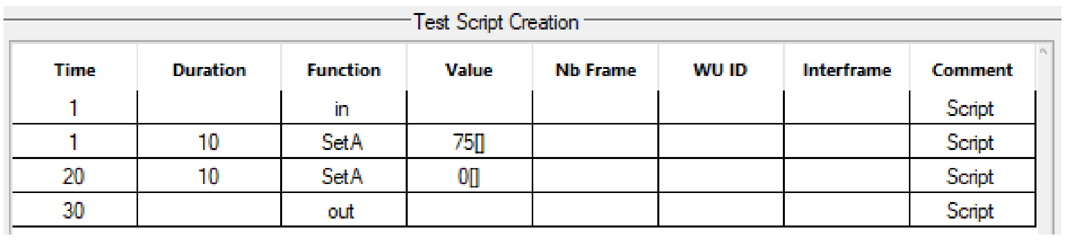
h. Label

Command	**Label**[1]
Interface	
Mandatory parameters	Time [s] Label name
Description	Delimits the period of analysis. E.g. From the ***Label 1*** at ***Time 1*** to the ***Label 2*** at ***Time 2***.

[1] Label is not a function that will command the test bench or ANum. Labels are tags necessary to analyze the test.

3. Additional features

Once all the steps have been added, you get a table like the Figure 7, and several features are available.

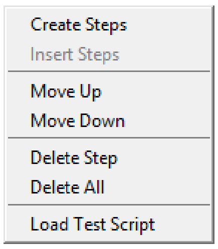

Test Script Creation

Time	Duration	Function	Value	Nb Frame	WU ID	Interframe	Comment
1		in					Script
1	10	SetA	75[]				Script
20	10	SetA	0[]				Script
30		out					Script

Figure 7. Example of a Test Script table

a. Modifying the table (step positions, load existing test)

As we have seen in the *Create a new command* part, a right click on the table gives you access to additional features:

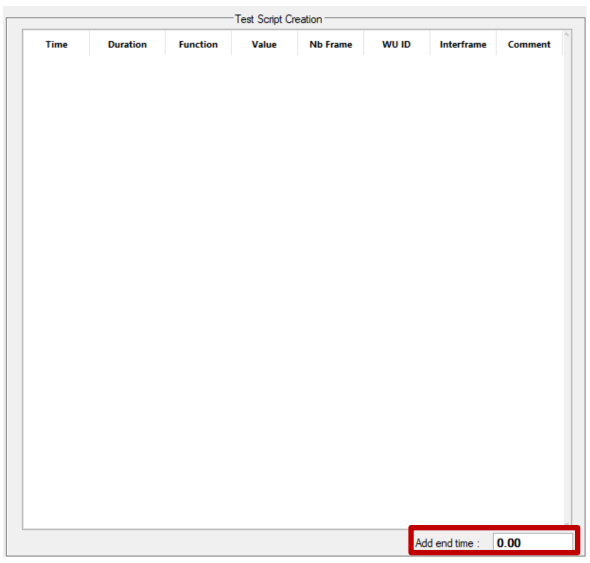


- *Move Up* and *Move Down*: change the position of the selected step[1].
- *Delete Step*: delete the selected step.
- *Delete All*: delete all steps of the table.
- *Load Test Script*: load the test script from an existing Test Case (.xlsx).

b. Adding an extra end time to the test

Moreover, you can add some extra time to your test in the area shown below, because sometimes, the WFC continue to send some frames after the end of the test script.

The test will not end at the time indicated in this field, but at the total time of the script plus the extra time.



[1] Changing the position of the step does not affect the test execution order. Only the time indicated in the **Time** field is considered.

4. Saving the Test script

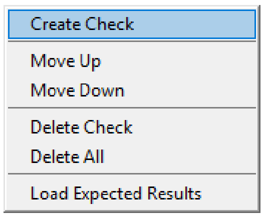
Once the Test script is created, you can create the Expected Results script too or save only the Test script. In this case, the Expected Results section will be empty.

To save the Test script, check the Saving the Test Case section.

B. Expected Results creation

Located at the right side of the window, this panel allows you to create the Expected Results script. The Expected Results script contains all the check functions (CheckNbBursts, CheckNoRF, CheckCompareAcc…) that will allow, once the execution is finished, to analyze the test and to give you the status PASSED or FAILED.

1. Create a new check function



A right click on the table gives you access to several features.
Click on ***Create Check*** to add a Check function to your table.

A new window titled "Expected results" appears, as presented below in the Figure 8.

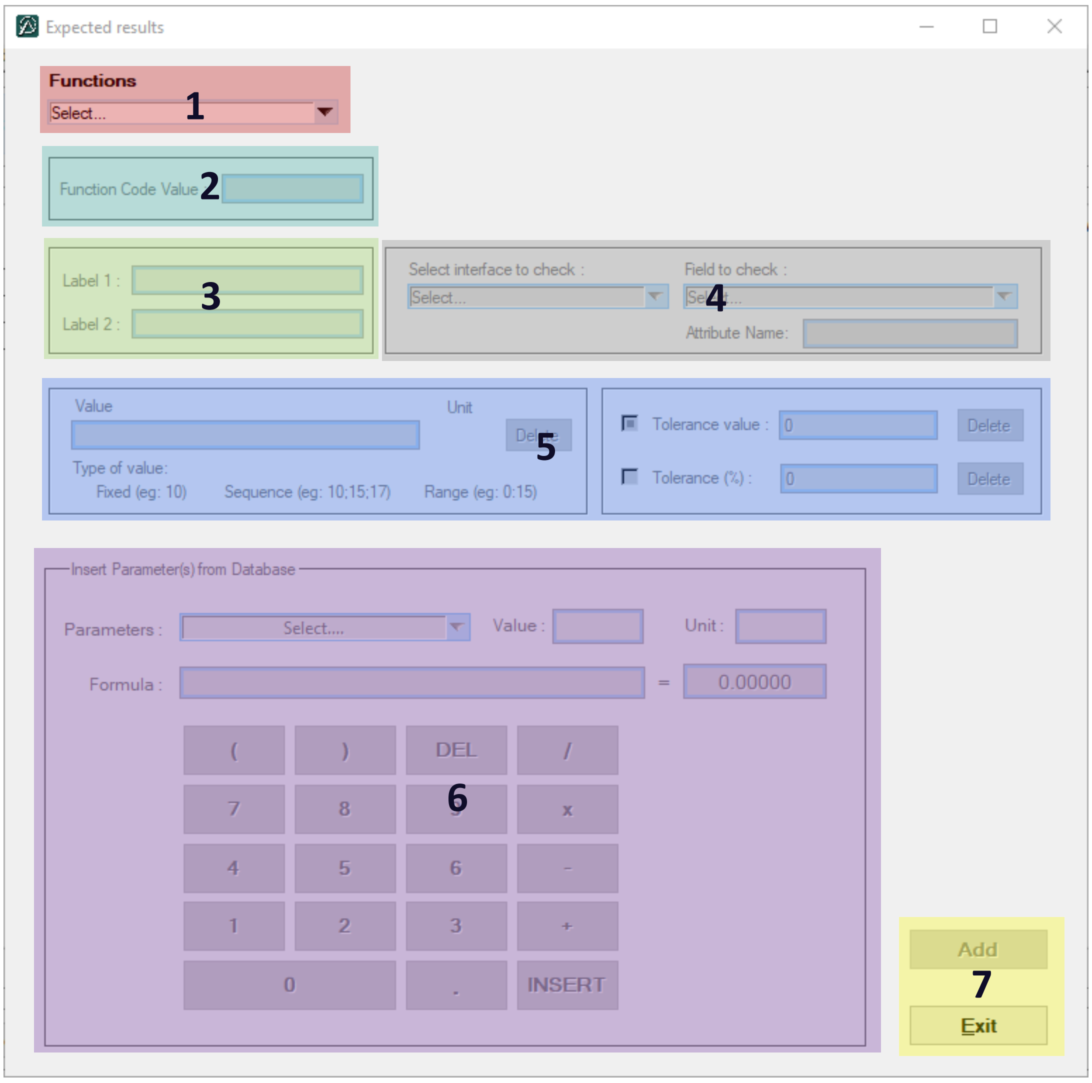


Figure 8. Expected Results window

The window is divided as follow:

(1) **Check functions selection:** this contextual menu allows you to choose the check function: (CheckTimingInterBursts, CheckTimingInterFrames, CheckFieldValue…). The various check functions are detailed in the *Check functions details* section.

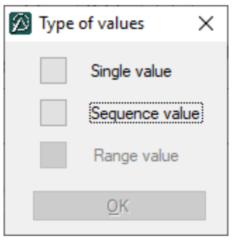
(2) **Function Code Value:** this text field allows you to indicate the Function Code value. The received frames contain a number named Function Code, designating the mode in which the sensor is. By indicating the Function Code value in this text field, you will filter the frames received by keeping only the frames with this value. To enter a value, you need to use the numeric keypad presented below.

(3) **Labels:** this panel all allows you to indicate the two labels[1] between which the analysis will be done. The names of the labels must correspond to those defined in the Test script.

(4) **Interface and field to check:** this panel allows you to indicate the RF frame format (Standard_frm, Diag_frm ...) and the field to check.
The frames received during a test may have a different interface[2]. This "Interface to check" contextual menu is used to filter the frames received according to the chosen interface.
The "Field to check" contextual menu is used to check the existence of a parameter in the log file during the analysis. The parameter is specific to each interface.

Note: *when you configure the CheckFieldValue function, this panel allows you to choose the parameter to check.*

(5) **Value and tolerances:** this panel allows you to indicate the expected value (timing, acceleration pressure, function code value…) and their its tolerance for the check function. The value depends of each check function.

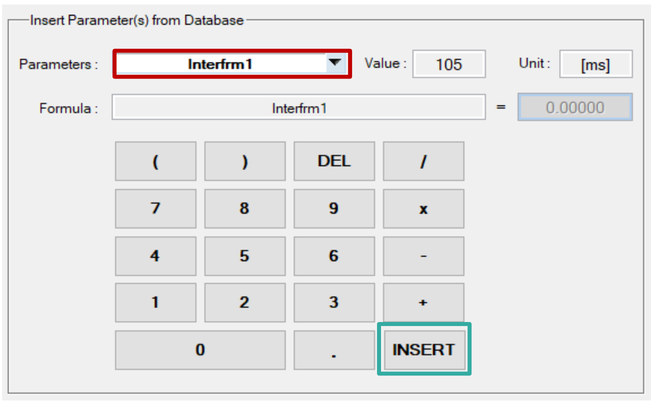


To enter a value, you must click on the Value field and first choose the type of value (Single, Sequence or Range value) as presented here.
Then, you need to use the numeric keypad presented below.

To enter a tolerance[3] (value or percentage), you need to use the numeric keypad too.
The unit must be the same for the value and tolerance.

[1] The CheckTimingFirstRF functions needs only one label (Label 1).
[2] The interfaces are defined in the Database.
[3] You can search for the exact value indicating a 0 value in the tolerance field.

(6) **Numeric keypad:** the numeric keypad, shown below in the Figure 9, allows you to enter the value of your choice (Function Code Value, Value, Tolerances…).


Insert Parameter(s) from Database

Parameters : Interfrm1 Value : 105 Unit : [ms]

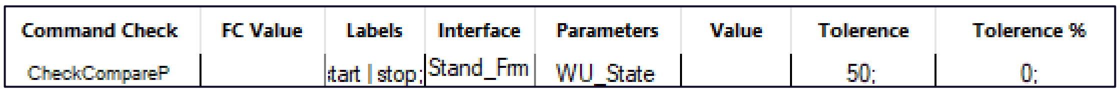
Formula : Interfrm1 = 0.00000

Figure 9. Numeric keypad for Expected Results creation

The value can be a numeric value (e.g. “102”, “10”) or a formula using parameters from the Database. You can choose a specific parameter from the Database clicking on the contextual menu titled “**Parameters**” The unit and the value are displayed next to the parameter’s name.

To validate the parameter value, click on the **INSERT** button.

(7) **Add and Exit buttons:** once the check function has been detailed, you can add it to the table using the button **Add**. A new line like shown below is added to the table:



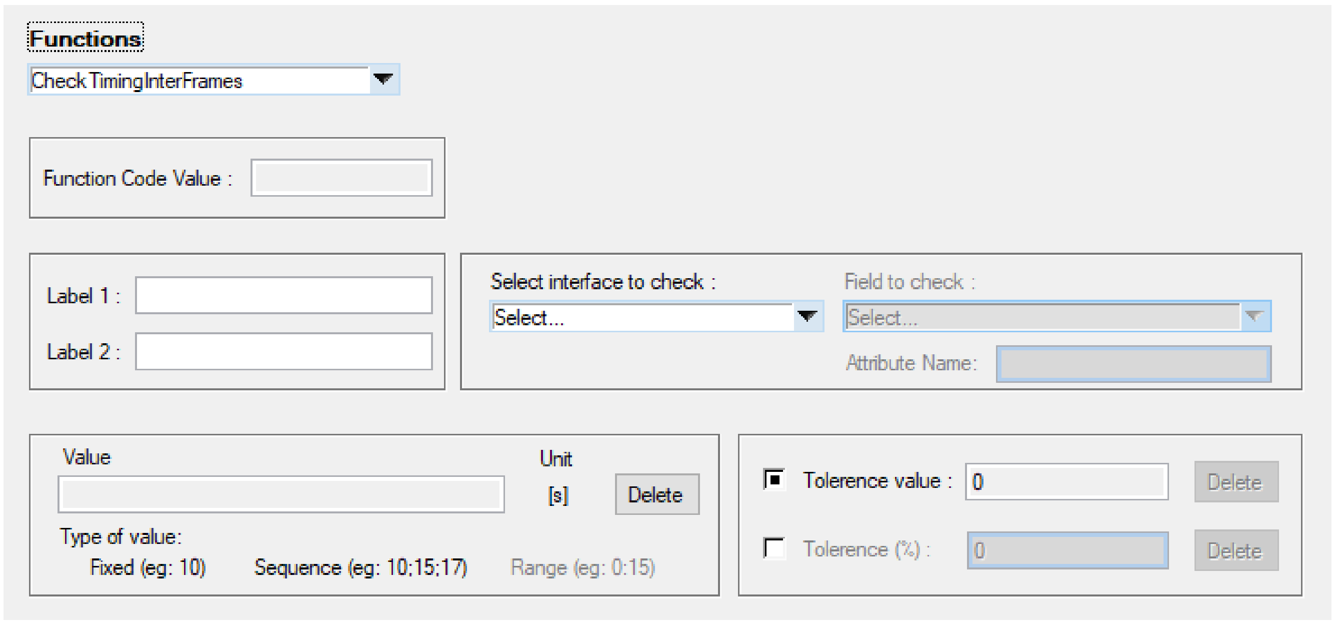
Command Check	FC Value	Labels	Interface	Parameters	Value	Tolerence	Tolerence %
CheckCompareP		tart | stop;	Stand_Frm	WU_State		50;	0;

You can quit the “Expected results” window at any time with the button **Exit**.

2. Check functions details

As we have seen above, the Expected results window allows you to configure various Check functions. The Figures below detail some important information.

a. CheckTimingInterFrames


Functions: Check TimingInterFrames

Function Code Value :

Label 1 :

Label 2 :

Select interface to check : Select...

Field to check : Select...

Attribute Name:

Value

Unit: [s]

Type of value: Fixed (eg: 10) Sequence (eg: 10;15;17) Range (eg: 0:15)

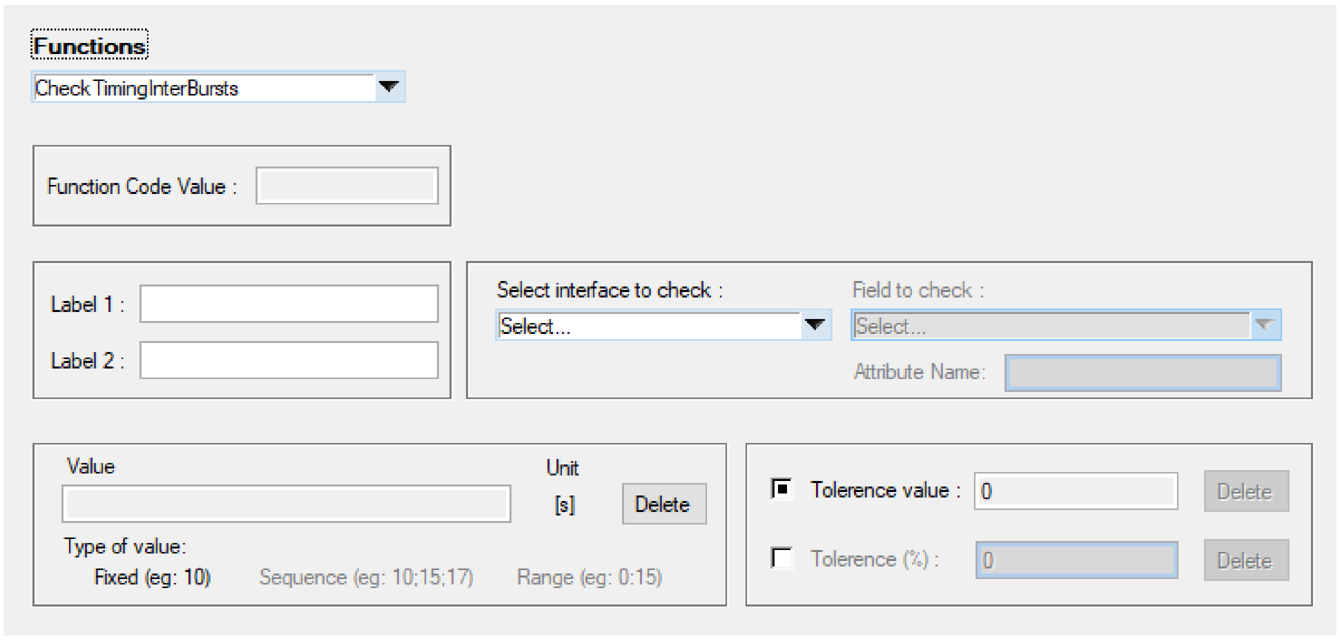
Tolerence value : 0

Tolerence (%) : 0

Purpose		
This function allows to check the timing between two frames.		
Mandatory fields	**Type of values**	**PASSED conditions**
Function Code value Label 1 and Label 2 Interface Field to check Value Tolerance (value or percentage)	Unit: [s] Fixed Range	Pre-conditions: - Function Code All the interframes between [**Value** +/- **Tolerance**].
Analysis		
First, the function filters the received frames keeping only those between the two labels and with an interface[1] and a FC value identical to those given as parameter. Then, the function checks, for every burst[2], if all timings between frames belong to the interval defined by the value(s) given as parameter, plus/minus tolerance.		

[1] The function checks if the "Field to check" linked to the Interface is an existing column in the log file.
[2] Two frames belong to the same burst if their interframe timing is below the threshold time (*Interframe/Interburst Threshold* value stored in the Database).

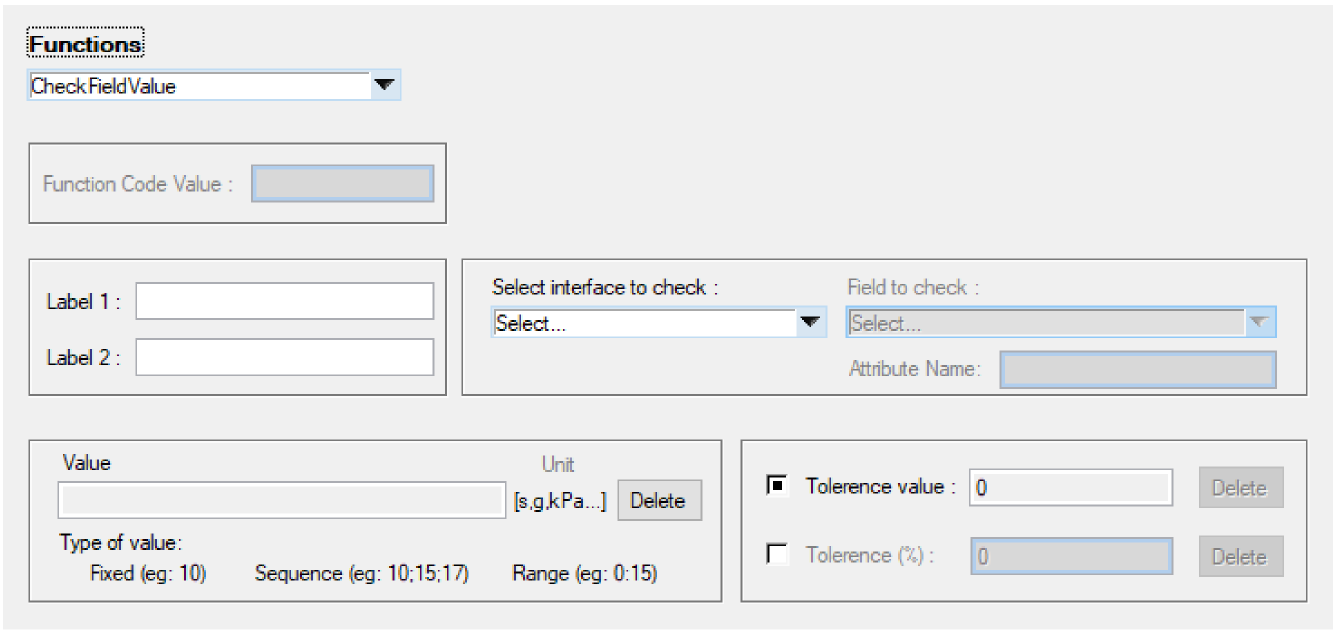
b. CheckTimingInterBursts



Purpose		
This function allows to check the timing between two bursts.		
Mandatory fields	**Type of values**	**PASSED conditions**
Function Code value Label 1 and Label 2 Interface Field to check Value Tolerance (value or percentage)	Unit: [s] Fixed	Pre-conditions: - Function Code All the interbursts between [**Value** +/- **Tolerance**].
Analysis		
First, the function filters the received frames keeping only those between the two labels and with an interface[1] and a FC value identical to those given as parameter. Then, the function checks if all timings between bursts[2] belong to the interval defined by the value(s) given as parameter, plus/minus tolerance.		

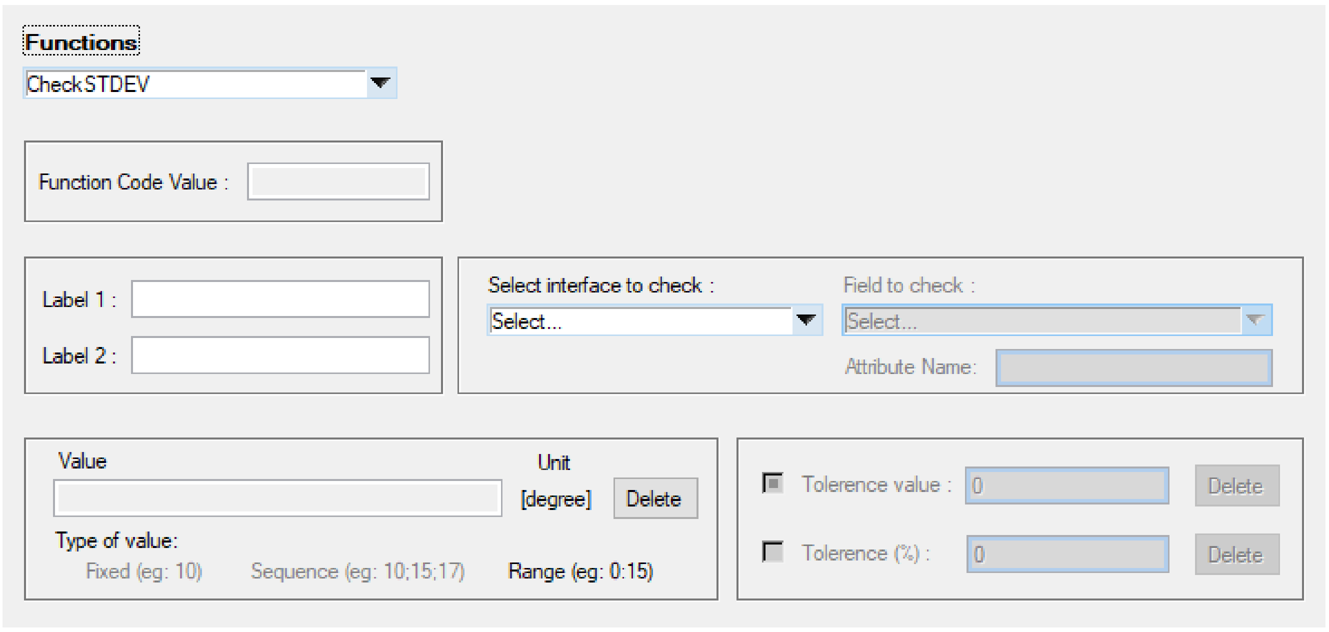
[1] The function checks if the “Field to check” linked to the Interface is an existing column in the log file.
[2] Two frames do not belong to the same burst if their interframe timing is higher than the threshold time (*Interframe/Interburst Threshold* value stored in the Database).

c. CheckFieldValue


Functions
CheckFieldValue
Function Code Value :
Label 1 :
Label 2 :
Select interface to check : Select...
Field to check : Select...
Attribute Name:
Value
Unit [s,g,kPa...] Delete
Type of value:
Fixed (eg: 10) Sequence (eg: 10;15;17) Range (eg: 0:15)
[✓ Tolerence value :] 0 Delete
Tolerence (%) : 0 Delete

Purpose		
This function allows to check the value of a specific field.		
Mandatory fields	**Type of values**	**PASSED conditions**
Label 1 and Label 2 Interface Field to check Value Tolerance (value or percentage)	Unit: [s, g, kPa…] Fixed Sequence Range	All the field values between [**Value** +/- **Tolerance**].
Analysis		
First, the function filters the received frames keeping only those between the two labels. Then, the function checks for all frames if the received value of the “Field to check” given as parameter belongs to the interval defined by the value(s) given as parameter, plus/minus tolerance.		

d. CheckSTDEV


Functions

CheckSTDEV

Function Code Value :

Label 1 :

Label 2 :

Select interface to check : Select...

Field to check : Select...

Attribute Name:

Value

Unit [degree]

Type of value: Fixed (eg: 10) Sequence (eg: 10;15;17) Range (eg: 0:15)

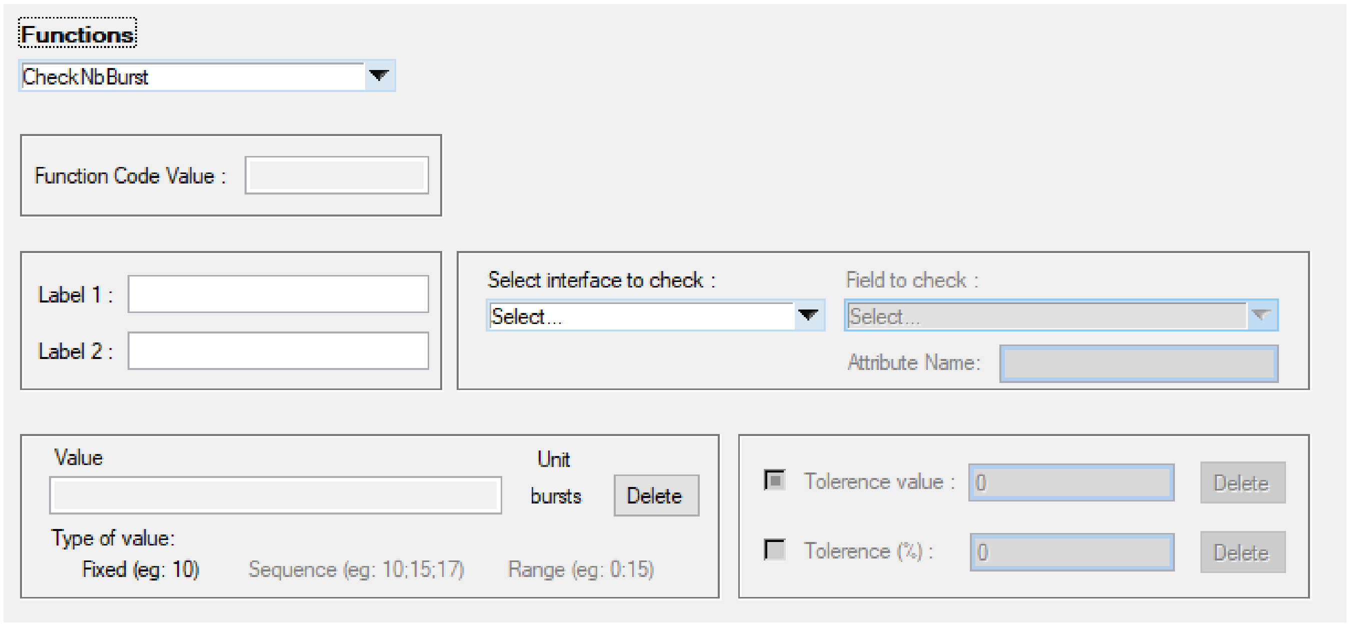
Tolerence value : 0

Tolerence (%) : 0

Purpose		
This function allows to check the efficiency of the LSE feature.		
Mandatory fields	**Type of values**	**PASSED conditions**
Function Code value Label 1 and Label 2 Interface Field to check Value	Unit: [degree] Range	Pre-conditions: - Function Code - Number of populations Standard deviation * 3 between [**Range value**].
Analysis		
First, the function filters the received frames keeping only those - between the two labels - with an interface[1] and a FC value identical to those given as parameter - corresponding to the first frame[2] of the burst - with an *Angle_detection* field equal to 1. Then, the function checks if the number of populations is equal to the one expected[3] and checks if every population has a standard deviation multiply by three between the range given as parameter. Finally, the function checks if the merged population[4] has a standard deviation multiply by three belongs to the interval defined by the range given as parameter. If the number of synchronized frames is below than 9, the standard deviation will be calculated but the status will be FAILED.		

[1] The function checks if the “Field to check” linked to the Interface is an existing column in the log file.
[2] The number of the first frame corresponds to the *Frame_number* field and must be equal to the *Frame_number* value stored in the Database.
[3] The number of populations expected is stored in the Database (*Number of emission angles for LSE*).
[4] The n populations are plotted on a single area to check the orthogonality of the populations (using mod 360).

e. CheckNbBursts


Functions

CheckNbBurst

Function Code Value :

Label 1 :

Label 2 :

Select interface to check : Select...

Field to check : Select...

Attribute Name:

Value

Unit

bursts

Delete

Type of value:

Fixed (eg: 10) Sequence (eg: 10;15;17) Range (eg: 0:15)

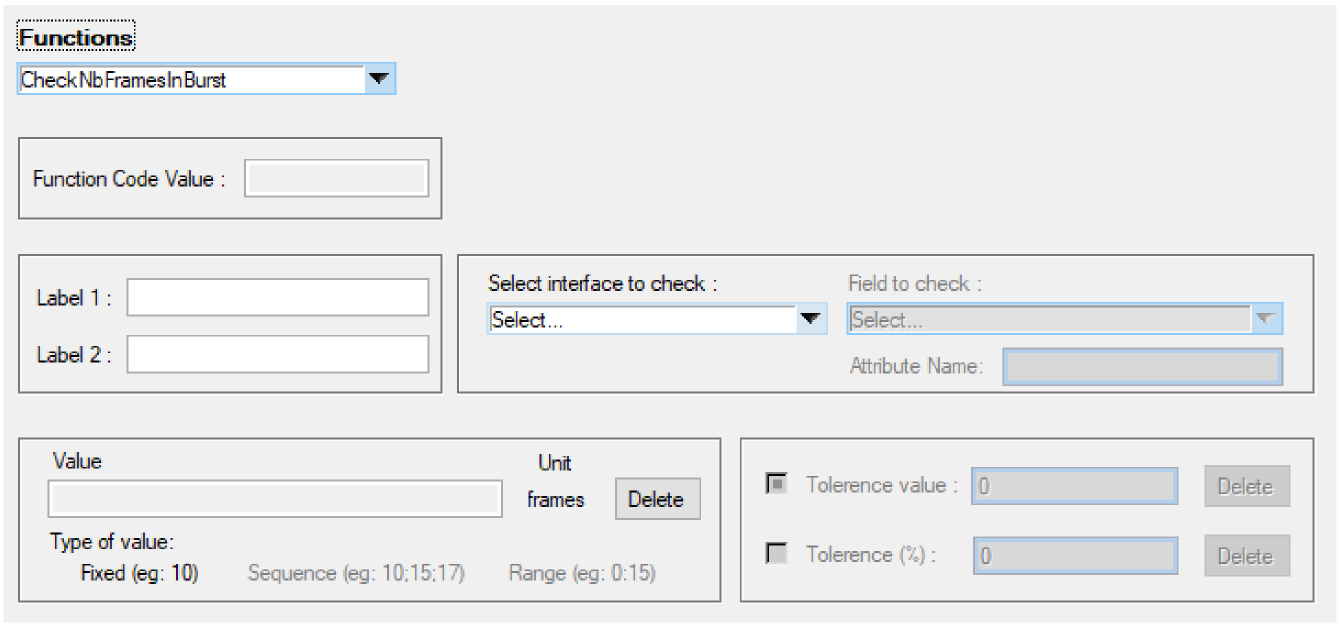
Tolerence value : 0 Delete

Tolerence (%) : 0 Delete

Purpose		
This function allows to check the number of bursts sent during the test.		
Mandatory fields	**Type of values**	**PASSED conditions**
Function Code value Label 1 and Label 2 Interface Field to check Value	Unit: [number] Fixed	Pre-conditions: - Function Code Number of bursts = **Value**.
Analysis		
First, the function filters the received frames keeping only those - between the two labels - with an interface[1] and a FC value identical to those given as parameter. Then, the function counts the number of bursts[2] and compares the number with the value given as parameter. The number of bursts counted must be exactly equal to the value given as parameter.		

[1] The function checks if the "Field to check" linked to the Interface is an existing column in the log file.
[2] Two frames do not belong to the same burst if their interframe timing is higher than the threshold time (*Interframe/Interburst Threshold* value stored in the Database).

f. CheckNbFramesInBurst


Functions

CheckNbFramesInBurst

Function Code Value :

Label 1 :

Label 2 :

Select interface to check : Select...

Field to check : Select...

Attribute Name:

Value

Unit: frames

Type of value: Fixed (eg: 10) Sequence (eg: 10;15;17) Range (eg: 0:15)

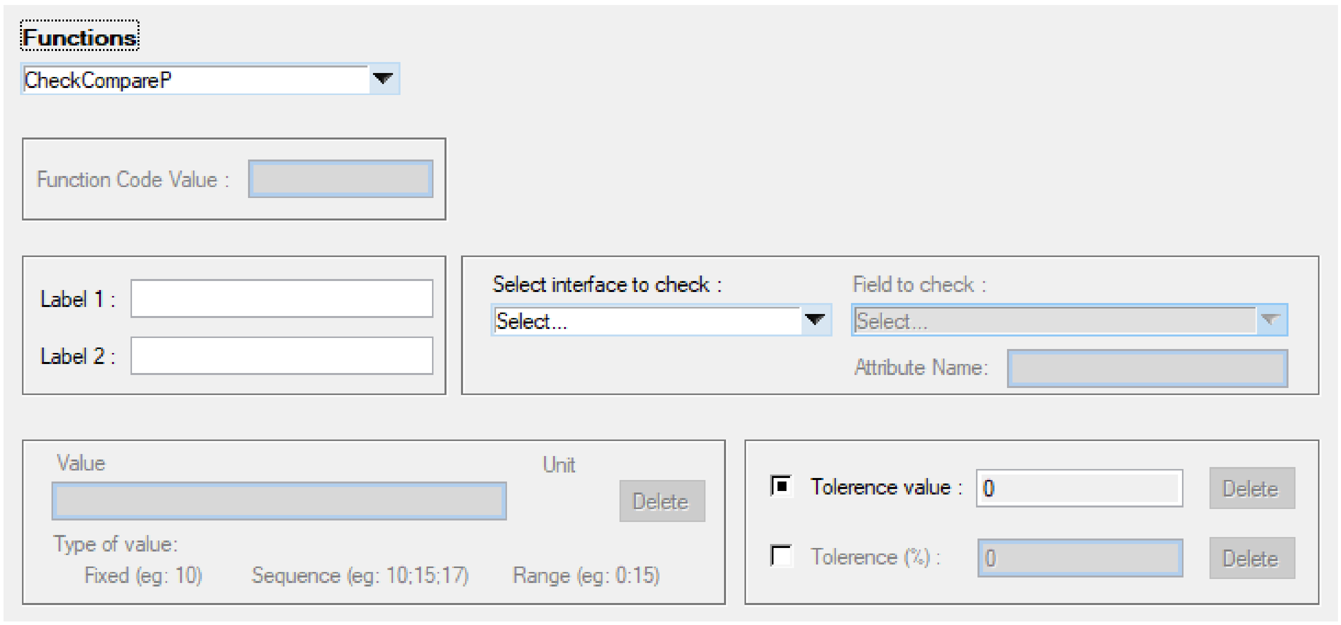
Tolerence value : 0

Tolerence (%) : 0

Purpose		
This function allows to check the number of frames in the first burst sent.		
Mandatory fields	**Type of values**	**PASSED conditions**
Function Code value Label 1 and Label 2 Interface Field to check Value	Unit: [number] Fixed	Pre-conditions: - Function Code Number of frames in first burst = **Value**.
Analysis		
First, the function filters the received frames keeping only those - between the two labels - with an interface[1] and a FC value identical to those given as parameter. Then, the function counts the number of frames in the first burst[2] and compares the number with the value given as parameter. The number of frames counted must be exactly equal to the value given as parameter.		

[1] The function checks if the “Field to check” linked to the Interface is an existing column in the log file.
[2] Two frames belong to the same burst if their interframe timing is below the threshold time (*Interframe/Interburst Threshold* value stored in the Database).

g. CheckCompareP



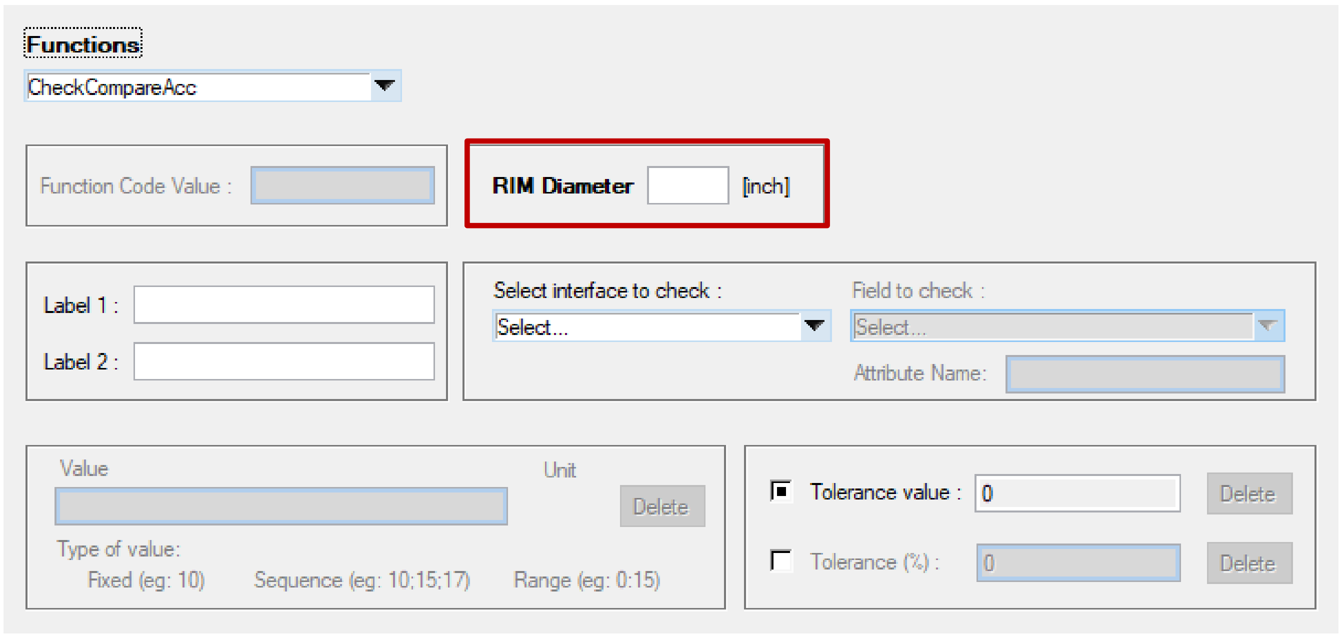
Purpose		
This function allows to check the accuracy of the pressure values sent by the WFC.		
Mandatory fields	**Type of values**	**PASSED conditions**
Label 1 and Label 2 Interface Field to check Tolerance (value or percentage)		All the pressure values between [**Value** +/- **Tolerance**].
Analysis		
First, the function filters the received frames keeping only those - between the two labels - with an interface[1] identical to the one given as parameter. Then, the function checks for every frame if the pressure value sent by the WFC[2] (from the *Pressure* field in log file) belongs to the interval defined by the reference pressure[3] value acquired by the bench plus/minus tolerance.		

[1] The function checks if the “Field to check” linked to the Interface is an existing column in the log file.

[2] The pressure value captured by the sensor is stored in the *Pressure* field from the log file.

[3] The reference pressure value is calculated by interpolation between two other known pressure values (from the bench) in order to guess the pressure value at the emission time of the frame.

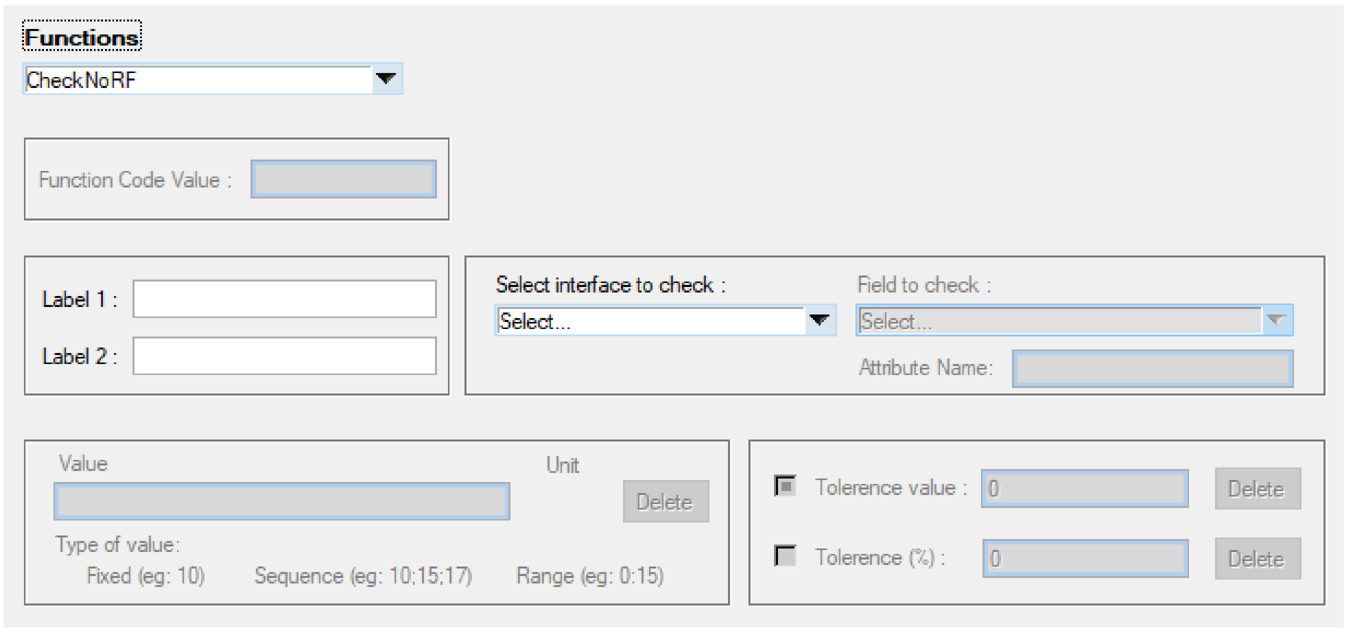
h. CheckCompareAcc



Purpose		
This function allows to check the accuracy of the acceleration values captured and sent by the WFC.		
Mandatory fields	**Type of values**	**PASSED conditions**
Label 1 and Label 2 Interface Field to check Tolerance (value or percentage) RIM Diameter		All the acceleration values between [**Value** +/- **Tolerance**].
Analysis		
First, the function filters the received frames keeping only those - between the two labels - with an interface[1] identical to the one given as parameter. Then, the function checks for every frame if the acceleration value sent by the WFC[2] belongs to the interval defined by the reference acceleration[3] value acquired by the bench plus/minus tolerance.		

[1] The function checks if the "Field to check" linked to the Interface is an existing column in the log file.
[2] The angular speed value captured by the WFC is stored in the *ASpeed* field from the log file and converted into acceleration (g) thanks to the RIM diameter given as parameter.
[3] The reference pressure value is stored in the *Acceleration* field from the log file.

i. CheckNoRF


Functions

CheckNoRF

Function Code Value :

Label 1 :

Label 2 :

Select interface to check : Select...

Field to check : Select...

Attribute Name:

Value

Unit

Delete

Type of value:

Fixed (eg: 10) Sequence (eg: 10;15;17) Range (eg: 0:15)

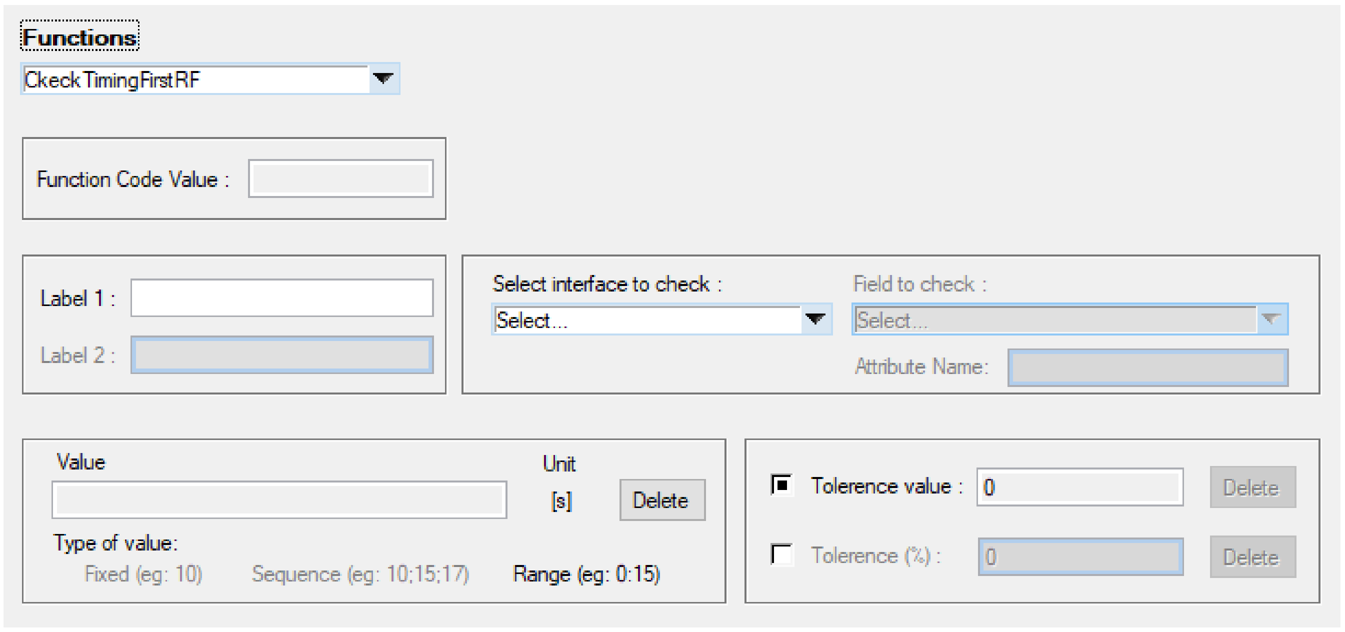
Tolerence value : 0 Delete

Tolerence (%) : 0 Delete

Purpose		
This function allows to check the accuracy of the pressure values captured and sent by the WFC.		
Mandatory fields	**Type of values**	**PASSED conditions**
Label 1 and Label 2 Interface Field to check		No frame received between **Labels**.
Analysis		
The function checks the absence of frame with the interface[1] given as parameter between labels. Note: the function does not detect frames with an interface other than the one indicated as parameter.		

[1] The function checks if the "Field to check" linked to the Interface is an existing column in the log file.

j. CheckTimingFirstRF


Functions

Ckeck TimingFirstRF

Function Code Value :

Label 1 :

Label 2 :

Select interface to check : Select...

Field to check : Select...

Attribute Name:

Value

Unit [s]

Type of value: Fixed (eg: 10) Sequence (eg: 10;15;17) Range (eg: 0:15)

Tolerence value : 0

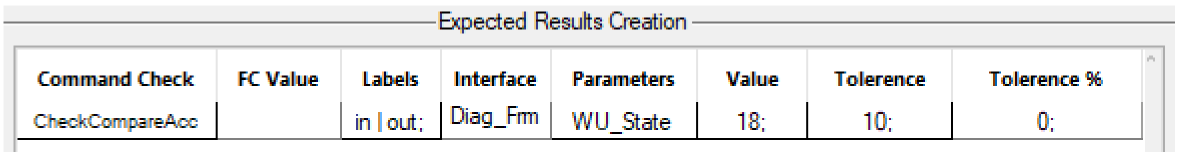
Tolerence (%) : 0

Purpose		
This function allows to check the timing to receive the first RF.		
Mandatory fields	**Type of values**	**PASSED conditions**
Function Code value Label 1 Interface Field to check Value	Unit: [s] Range	Pre-conditions: - Function Code Timing "Label 1 - first RF" +/- **Tolerance** between [**Range Value**]
Analysis		
First, the function filters the received frames keeping only those with an interface[1] and a FC value identical to those given as parameter. Then, the function checks if the timing between the Label 1 and the first RF belongs to the interval defined by the range given as parameter.		

[1] The function checks if the "Field to check" linked to the Interface is an existing column in the log file.

3. Additional features

Once all the steps have been added, you get a table like the Figure 10, and several features are available.



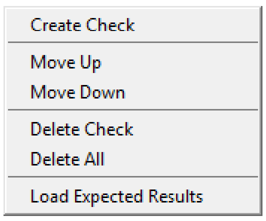
Expected Results Creation

Command Check	FC Value	Labels	Interface	Parameters	Value	Tolerence	Tolerence %
CheckCompareAcc		in | out;	Diag_Frm	WU_State	18;	10;	0;

Figure 10. Example of a Test Script table

a. Modifying the table (step positions, load existing test)

As we have seen in the *Create a new check function* section, a right click on the table gives you access to additional features:



- *Move Up* and *Move Down*: changes the position of the check function[1].
- *Delete Step*: deletes the selected check function.
- *Delete All*: deletes all check functions of the table.
- *Load Expected Results*: loads the Expected Results script from an existing Test Case (.xlsx).

4. Saving the Expected Results script

To save the Expected Results script, check the Saving the Test Case section.

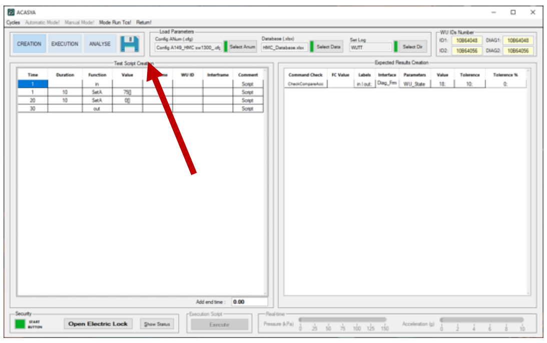
[1] Changing the position of the step does not affect the test execution order. Only the time indicated in the **Time** field is considered.

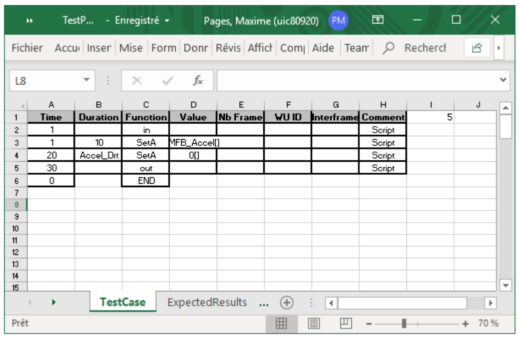
C. Saving the Test Case

As a reminder, the Test Case contains the Test script and the Expected Results script.

To save your Test Case, click on the **Save Button** located in the **Mode Selection panel** at the top of your window. Then, a new window allows you to choose where to save the Test Case file.

Note: you can save a Test Case with an empty Expected Results table.



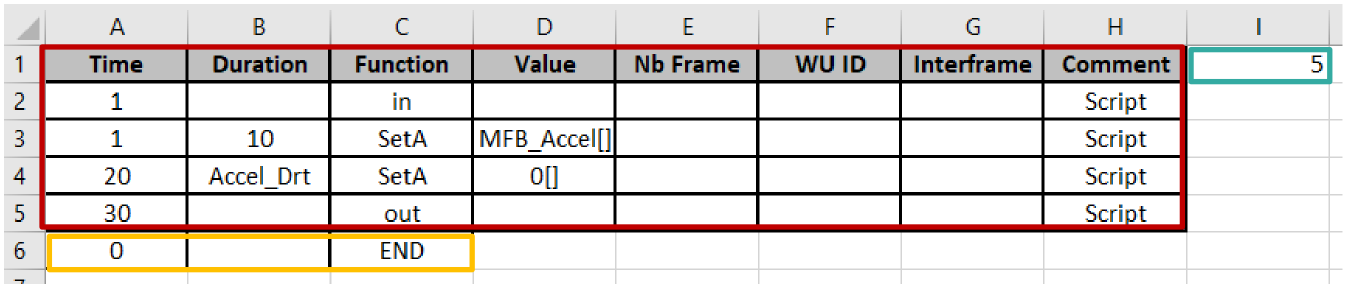


	A	B	C	D	E	F	G	H	I
1	Time	Duration	Function	Value	Nb Frame	WU ID	Interframe	Comment	5
2	1		in					Script	
3	1	10	SetA	MFB_Accel[]				Script	
4	20	Accel_Drt	SetA	0[]				Script	
5	30		out					Script	
6	0		END						

Saving the Test Case generates a **.xlsx file**, an **Excel table**. This Excel workbook contains two worksheets, one dedicated to the Test script and the other to Expected Results script.

a. Test script worksheet

The worksheet dedicated to the Test script is divided into 3 parts as shown below in the Figure 11:



	A	B	C	D	E	F	G	H	I
1	Time	Duration	Function	Value	Nb Frame	WU ID	Interframe	Comment	5
2	1		in					Script	
3	1	10	SetA	MFB_Accel[]				Script	
4	20	Accel_Drt	SetA	0[]				Script	
5	30		out					Script	
6	0		END						

Figure 11. Test script saved in Excel

Body of Test Script

These lines contain all the commands of the test previously created. The parameters are still present and will be translated during the execution.

End command

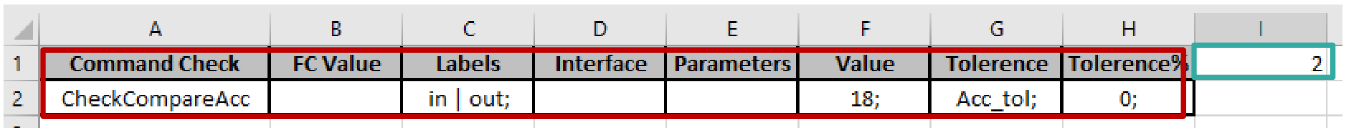
This line is always located after the last command of the test. The END is a tag that indicates the end of the test script. The value in the *Time* column indicates the total execution time of the test. Initially, this value is at zero or at the value indicated in the Add end time field. The final value is updated during the execution (at the very beginning).

Number of steps

This cell contains the number of rows minus the header line. It is useful during the Test Case execution.

b. Test script worksheet

The worksheet dedicated to the Test script is divided into 2 parts as shown below in the Figure 12:



	A	B	C	D	E	F	G	H	I
1	Command Check	FC Value	Labels	Interface	Parameters	Value	Tolerence	Tolerence%	2
2	CheckCompareAcc		in | out;			18;	Acc_tol;	0;	

Figure 12. Expected Results script saved in Excel

Body of Test Script

These lines contain all the commands of the test previously created. The parameters are still present and will be translated during the execution.

Number of steps

This cell contains the number of rows. It is useful during the Test Case execution.

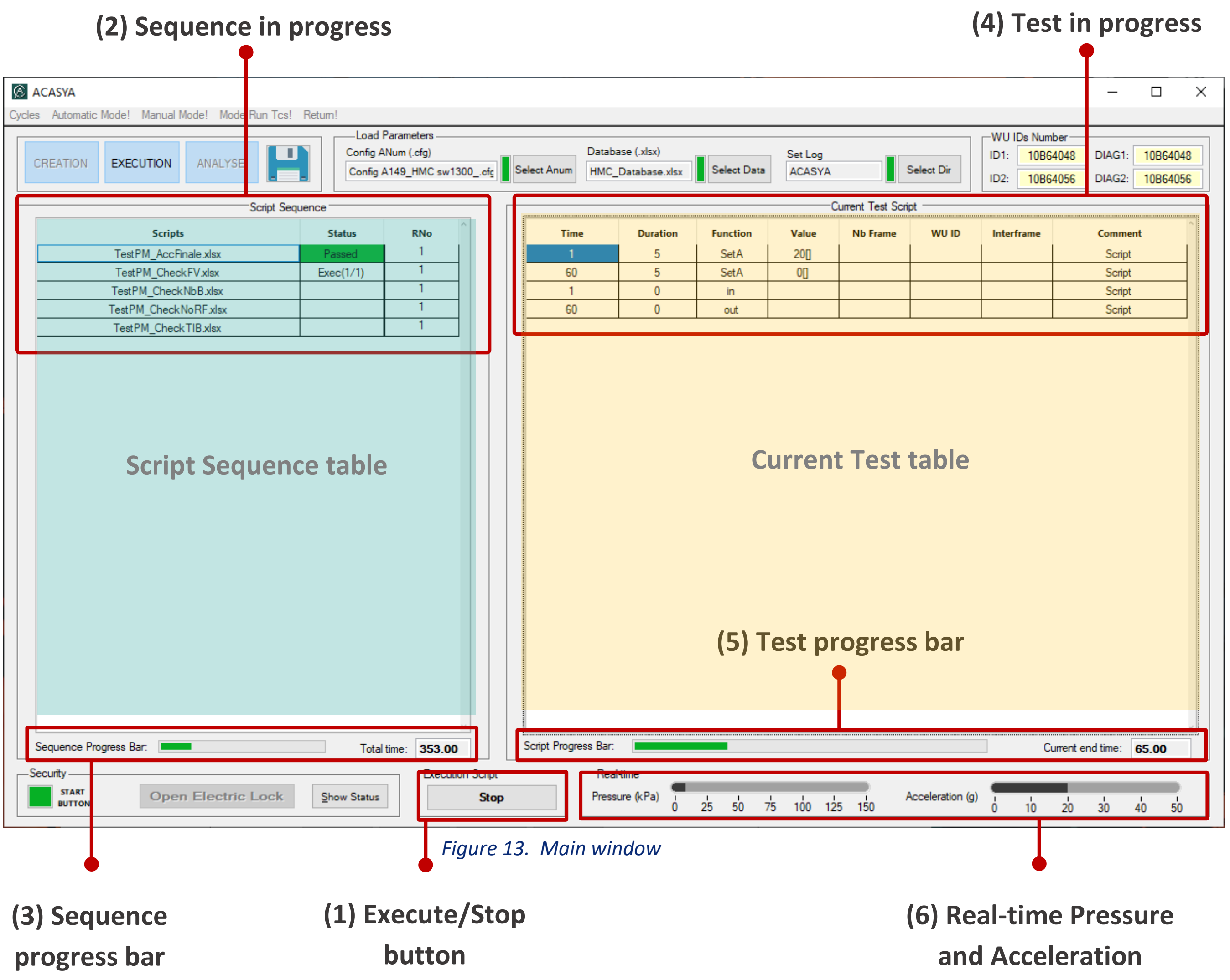
Chapter 3

EXECUTION MODE

III. EXECUTION MODE

A. IHM

In this mode you can execute a sequence of tests previously created. The window is organized as follow:



Scripts	Status	RNo
TestPM_AccFinale.xlsx	Passed	1
TestPM_CheckFV.xlsx	Exec(1/1)	1
TestPM_CheckNbB.xlsx		1
TestPM_CheckNoRF.xlsx		1
TestPM_CheckTIB.xlsx		1

Time	Duration	Function	Value	Nb Frame	WU ID	Interframe	Comment
1	5	SetA	20[]				Script
60	5	SetA	0[]				Script
1	0	in					Script
60	0	out					Script

Figure 13. Main window

(1) Execute/Stop button: button to start the test execution or to stop it at any time.

(2) Sequence in progress: table of the running sequence.

(3) Sequence progress bar: visual indicator to know the sequence progression.

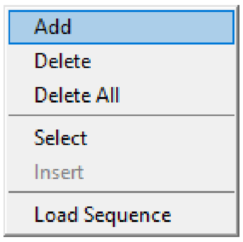
(4) Test in progress: table of the running test script.

(5) Test progress bar: visual indicator to know the test progression.

(6) Real-time Pressure and Acceleration: visual indicator to know the Pressure and Acceleration values in real-time during the test execution.

B. Executing a new sequence

1. Adding tests to your sequence

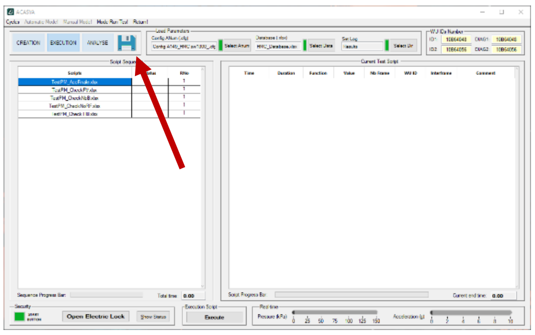


A right click on the Script Sequence Table gives you access to several features. Click on ***Add*** to add a Test Case to your table.

A new window appears and allows you to choose one or several Test Cases.

Note: *right clicking on the Script Sequence Table allows you to delete tests or to load an existing sequence (previously saved).*

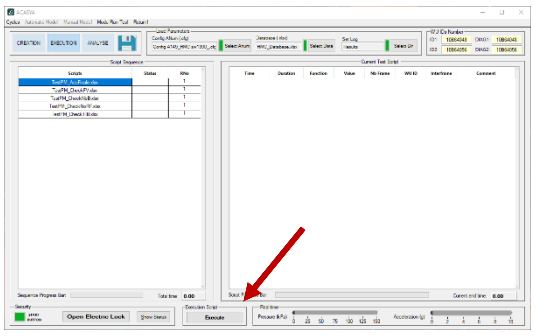
2. Starting the execution



Step 1: Once the tests are loaded, you must save the sequence in a .txt file.

To save your sequence, click on the **Save Button** located in the **Mode Selection panel** at the top of your window. Then, a new window allows you to choose where to save the sequence.

Note: *the application always checks if you have saved the sequence at the beginning of the execution.*



Step 2: Now, you can start your sequence.

To start your sequence, click on the **Execution Button** located at the bottom of your window.

A picture of a timer should appear and indicates that the execution is running.

First of all, ACASYA starts to translate every test loaded. This first step allows to translate a Test Case with parameters into a Test Case[1] with only values.

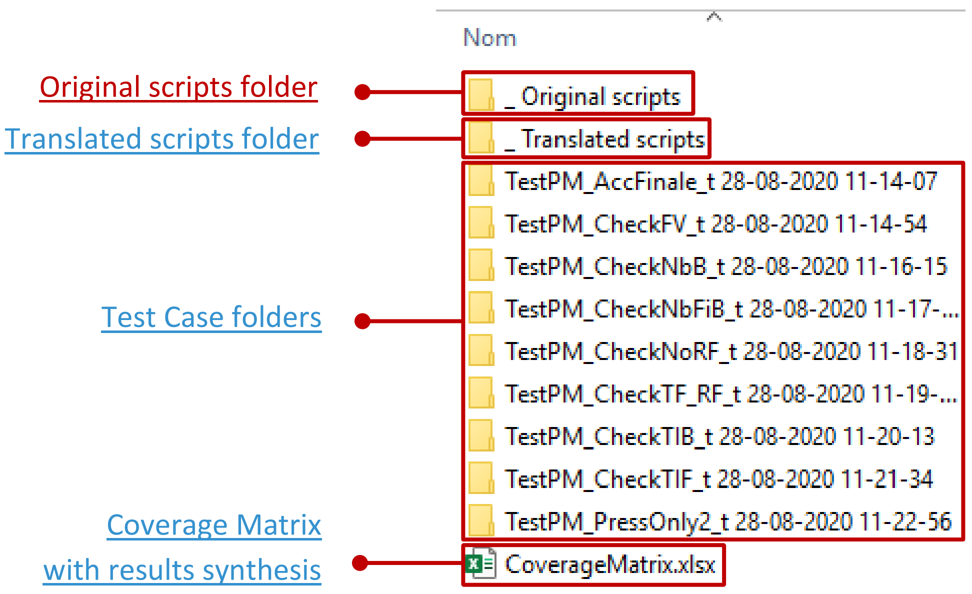
Then, Test Case scripts are executed and displayed in the Current Test Table one by one until the last.

[1] The translated Test Case scripts are stored in a folder named "_ Translated scripts" inside the sequence folder.

C. Analyzing the results

During the execution, ACASYA creates a sequence folder where tests, configuration files and results are stored. The sequence folder is stamped with the date and time, as presented here: Seq_28-08-20_11-13-46

Once the execution is complete, the sequence folder contains several files and subfolders as shown in the Figure 14:

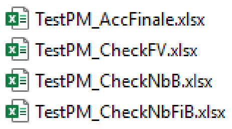

Original scripts folder
Translated scripts folder
Test Case folders
Coverage Matrix with results synthesis

Nom
_ Original scripts
_ Translated scripts
TestPM_AccFinale_t 28-08-2020 11-14-07
TestPM_CheckFV_t 28-08-2020 11-14-54
TestPM_CheckNbB_t 28-08-2020 11-16-15
TestPM_CheckNbFiB_t 28-08-2020 11-17-...
TestPM_CheckNoRF_t 28-08-2020 11-18-31
TestPM_CheckTF_RF_t 28-08-2020 11-19-...
TestPM_CheckTIB_t 28-08-2020 11-20-13
TestPM_CheckTIF_t 28-08-2020 11-21-34
TestPM_PressOnly2_t 28-08-2020 11-22-56
CoverageMatrix.xlsx

Figure 14. Sequence folder

1. Original scripts folder

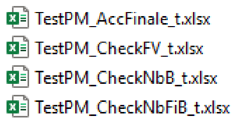
The **Original script folder** contains all the original Test Case scripts.
This folder allows you to easily access the scripts with the parameters.



TestPM_AccFinale.xlsx
TestPM_CheckFV.xlsx
TestPM_CheckNbB.xlsx
TestPM_CheckNbFiB.xlsx

2. Translated scripts folder

The **Translated script folder** contains all the translated Test Case scripts.
As have seen above, ACASYA translates at the very beginning all Test Case scripts with parameters into Test Case scripts with only value. This folder allows you to easily access the scripts with values only.

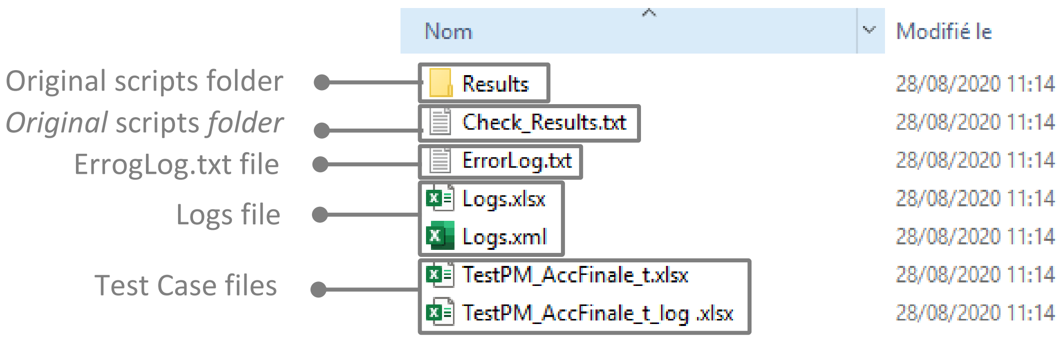


TestPM_AccFinale_t.xlsx
TestPM_CheckFV_t.xlsx
TestPM_CheckNbB_t.xlsx
TestPM_CheckNbFiB_t.xlsx

3. Test Case folders

The **Test Case folders** are very important folders because they contain the analysis results of each Test Case. There are as many folders as Test Case scripts.

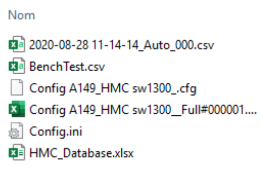
A Test Case folder contains several elements as shown in the Figure 15:



Nom	Modifié le
Results	28/08/2020 11:14
Check_Results.txt	28/08/2020 11:14
ErrorLog.txt	28/08/2020 11:14
Logs.xlsx	28/08/2020 11:14
Logs.xml	28/08/2020 11:14
TestPM_AccFinale_t.xlsx	28/08/2020 11:14
TestPM_AccFinale_t_log .xlsx	28/08/2020 11:14

Figure 15. Test Case folder

a. Results folder



Nom
2020-08-28 11-14-14_Auto_000.csv
BenchTest.csv
Config A149_HMC sw1300_.cfg
Config A149_HMC sw1300__Full#000001....
Config.ini
HMC_Database.xlsx

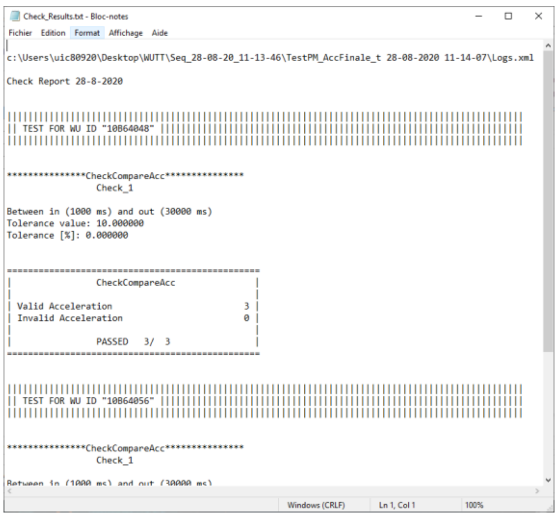
The Results folder contains several configuration files using by ACASYA during the execution such as the Database, the Config.ini file (with some environmental parameters), the ANum configuration, the original log file with received frames, etc.

b. Check_Results.txt file

The Check_Results.txt file is the most useful file of the Test Case folder. This text file contains the **analyse result** of the Test Case.

For each WFC, the text file indicates:
• the WU ID value and the Test Case name.
• the test parameters (labels, tolerances, FC value, etc.).
• the test conditions (expected value).
• rejected frames with wrong value.
• and whether the test is PASSED or FAILED.

You can find an example of a PASSED and FAILED Check_Results.txt in the Appendix.



```
c:\Users\uic80920\Desktop\WUTT\Seq_28-08-20_11-13-46\TestPM_AccFinale_t 28-08-2020 11-14-07\Logs.xml

Check Report 28-8-2020


||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
|| TEST FOR WU ID "10B64048" |||||||||||||||||||||||||||||||||||||||||||||||||||||||||
||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||


***************CheckCompareAcc***************
                 Check_1

Between in (1000 ms) and out (30000 ms)
Tolerance value: 10.000000
Tolerance [%]: 0.000000


================================================
|                CheckCompareAcc               |
|                                              |
| Valid Acceleration                         3 |
| Invalid Acceleration                       0 |
|                                              |
|                PASSED   3/  3                |
================================================


||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
|| TEST FOR WU ID "10B64056" |||||||||||||||||||||||||||||||||||||||||||||||||||||||||
||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||


***************CheckCompareAcc***************
                 Check_1

Between in (1000 ms) and out (30000 ms)
```

Figure 16. CheckResults.txt

c. ErrorLog.txt

The ErrorLog text file specifies the status of the Application during and after performing each test. The Error Log text file contains some information such as:

- the Test Case name.
- the Configuration File Parameters and Reading Status: Successful or Failed.
- the Test Case decoding errors.
- the Test Case Commands after decoding.
- and the Test Case Reading Status: Successful or Failed.

d. Logs files

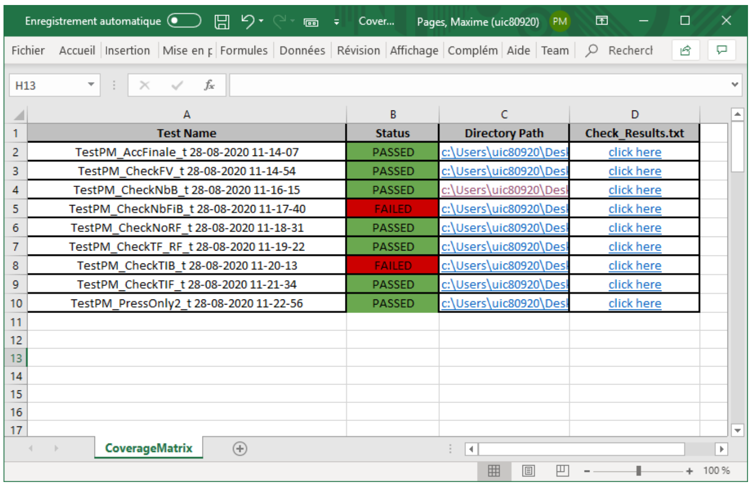
The Logs.xml file contains all the logs with frames received.
The Logs.xlsx file is the log processed by ACASYA during the analysis including labels.
These files are very useful for understanding where the errors are coming from.

e. Test Case files

These Test Case files is the Test Case scripts used and generated by ACASYA during the analyse.

4. Coverage Matrix with results synthesis

The CoverageMatrix.xlsx is a very useful file to analyze your sequence. This Excel workbook allows you to quickly see which tests are passed or failed and to easily access to the Test Case folders and the CheckResults.txt files.



Test Name	Status	Directory Path	Check_Results.txt
TestPM_AccFinale_t 28-08-2020 11-14-07	PASSED	c:\Users\uic80920\Des	click here
TestPM_CheckFV_t 28-08-2020 11-14-54	PASSED	c:\Users\uic80920\Des	click here
TestPM_CheckNbB_t 28-08-2020 11-16-15	PASSED	c:\Users\uic80920\Des	click here
TestPM_CheckNbFiB_t 28-08-2020 11-17-40	FAILED	c:\Users\uic80920\Des	click here
TestPM_CheckNoRF_t 28-08-2020 11-18-31	PASSED	c:\Users\uic80920\Des	click here
TestPM_CheckTF_RF_t 28-08-2020 11-19-22	PASSED	c:\Users\uic80920\Des	click here
TestPM_CheckTIB_t 28-08-2020 11-20-13	FAILED	c:\Users\uic80920\Des	click here
TestPM_CheckTIF_t 28-08-2020 11-21-34	PASSED	c:\Users\uic80920\Des	click here
TestPM_PressOnly2_t 28-08-2020 11-22-56	PASSED	c:\Users\uic80920\Des	click here

Figure 17. CoverageMatrix.xlsx

- Click on the hyperlinks of the “Directory Path” column to access to the Test Case folder.
- Click on the hyperlinks of the “Check_Results.txt” column to access to the Check_Results.txt file.

Chapter 4

ANALYSE MODE

IV. ANALYSE MODE

The Analyse Mode is the last mode added to ACASYA in the last update. This very useful mode allows you to analyse Log file from a previously performed test. It is a perfect mode to save time when you don't want to execute a test once again.

A. IHM

The main window is divided as follow:

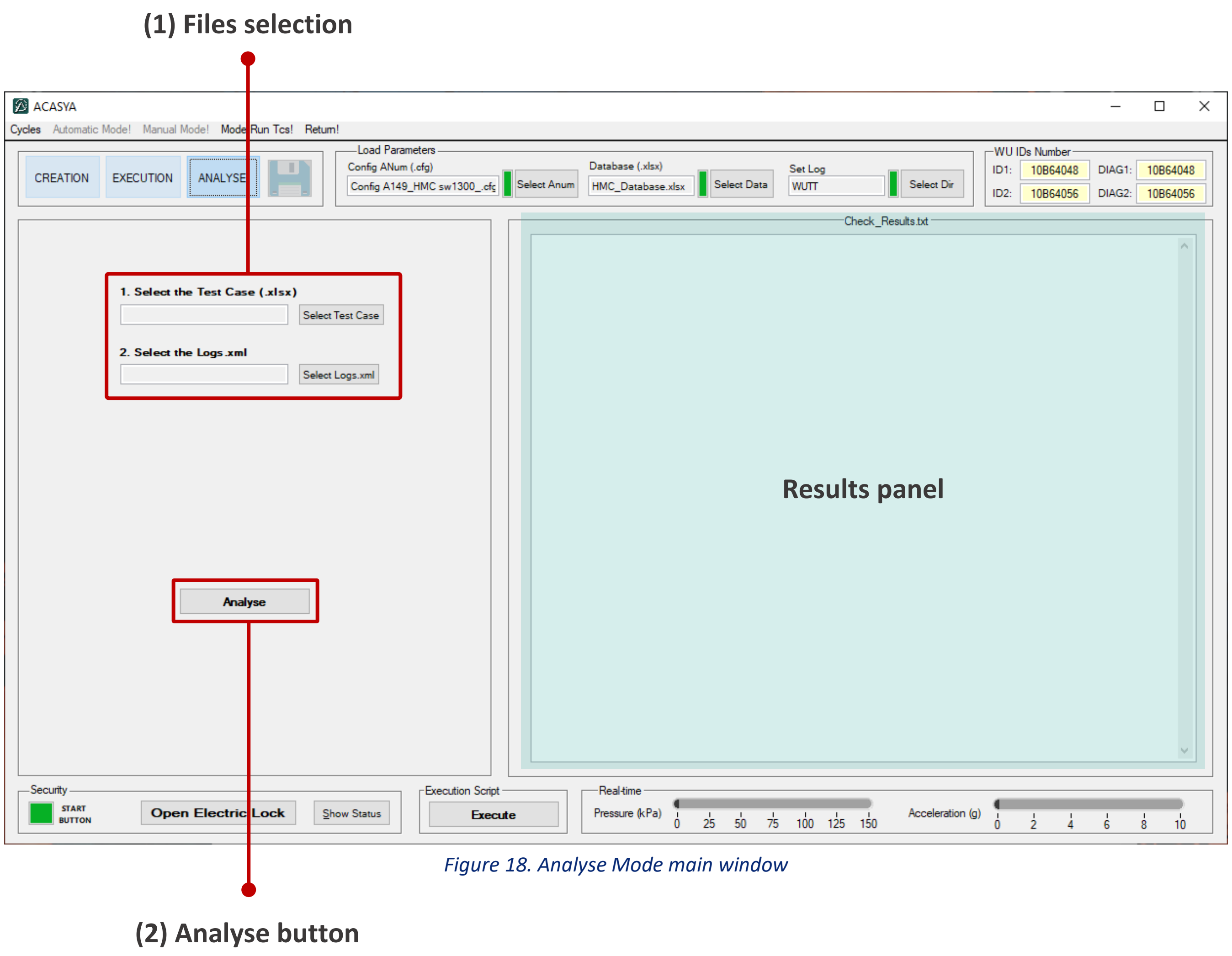


Figure 18. Analyse Mode main window

(1) Files selection: panel to select the Test Case including the Expected Results script (with check functions) and the Logs.xml file.

(2) Analyse button: button to start the analyse

B. Starting an analysis

To start an analysis, you need to indicate the Test Case including the Expected Results script (with check functions) and the Logs.xml file.

Step 1: Click on the *Select Test Case* button in the File Selection panel to choose the Test Case.

Step 2: Click on the *Select Logs.xml* button in the File Selection panel to choose the Logs.xml file.

Step 3: Once the Test Case and the Logs.xml file are loaded, click on the Analyse button to start the analysis.

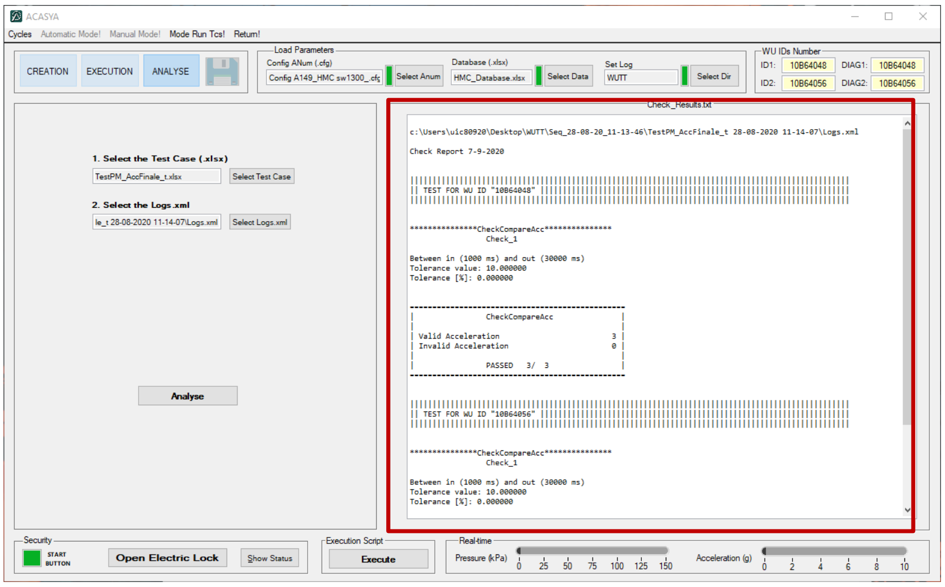
A picture of a timer should appear, and the analysis may take few seconds.

C. Analysis results

Once the analysis completed, ACASYA creates in your working folder a new Test Case folder. The Test Case folder is stamped with the date and time, as presented here: TestPM_CheckNbFiB_28-08-20_16-18-38

The Test Case folder contains the Test Case script and the Check_Results.txt, identical to the one presented in the *Check_Results.txt file* section.

In addition, the Results panel displays the Check_Results.txt file created during the analyse and allows you to quickly verify whether the test is passed or failed.



Config ANum (.cfg): Config A149_HMC sw1300_.cfg
Database (.xlsx): HMC_Database.xlsx
Set Log: WUTT

ID1: 10B64048 DIAG1: 10B64048
ID2: 10B64056 DIAG2: 10B64056

1. Select the Test Case (.xlsx)
TestPM_AccFinale_t.xlsx

2. Select the Logs.xml
le_t 28-08-2020 11-14-07\Logs.xml

```
c:\Users\uic80920\Desktop\WUTT\Seq_28-08-20_11-13-46\TestPM_AccFinale_t 28-08-2020 11-14-07\Logs.xml

Check Report 7-9-2020


|||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
|| TEST FOR WU ID "10B64048" ||||||||||||||||||||||||||||||||||||||||||||||||||||||
|||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||


***************CheckCompareAcc***************
                 Check_1

Between in (1000 ms) and out (30000 ms)
Tolerance value: 10.000000
Tolerance [%]: 0.000000


------------------------------------------------
|                CheckCompareAcc               |
|                                              |
| Valid Acceleration                         3 |
| Invalid Acceleration                       0 |
|                                              |
|                PASSED   3/  3                |
------------------------------------------------


|||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
|| TEST FOR WU ID "10B64056" ||||||||||||||||||||||||||||||||||||||||||||||||||||||
|||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||


***************CheckCompareAcc***************
                 Check_1

Between in (1000 ms) and out (30000 ms)
Tolerance value: 10.000000
Tolerance [%]: 0.000000
```

Figure 19. Analyse Results panel

Chapter 5

SECURITY AND MONITORING

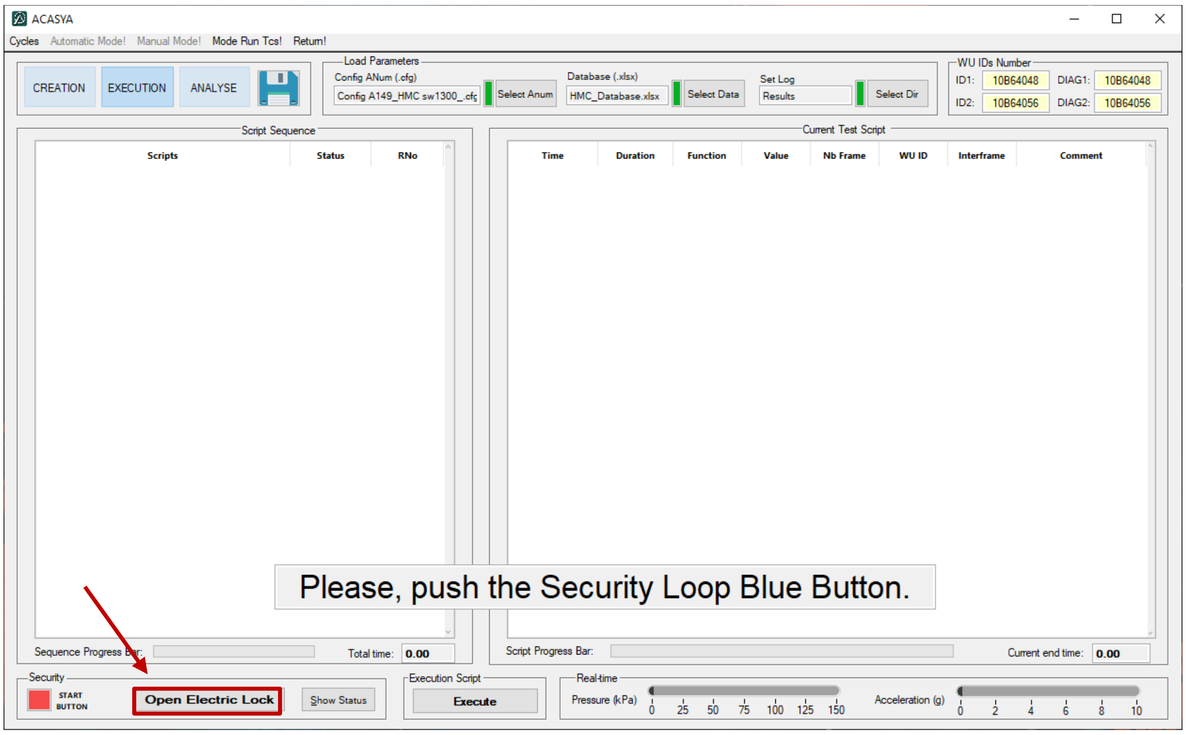
V. SECURITY AND MONITORING

During tests, ACASYA sends pressure and acceleration commands to the LSE test bench. Some dangerous behavior for the user can occur: the pressure and acceleration values are too high, the safety doors are not closed...

To avoid any dangerous behavior, ACASYA was designed with some safety mechanisms.

A. Electric Lock and Security Loop Blue button

If you want to have access to the WFCs, you need to open the Electric Locks clicking the **Open Electric Lock button**, presented below in the Figure 20:



Config ANum (.cfg): Config A149_HMC sw1300_.cfg
Database (.xlsx): HMC_Database.xlsx
Set Log: Results
ID1: 10B64048 DIAG1: 10B64048
ID2: 10B64056 DIAG2: 10B64056
Total time: 0.00
Current end time: 0.00

Please, push the Security Loop Blue Button.

Figure 20. Electric lock and Security Loop Blue Button

Once the Electric Locks are closed, click again on the same button to validate the closing of the doors. The message "*Please, push the Security Loop Blue Button*", as shown above, now indicates to press the Blue button located near to the LSE test bench in order to interact with the bench safely.

B. Monitoring and status panel

Clicking on the **Show Status button** allows you to display the Status window, as presented in the Figure 22.

This window presented in the Figure 22. Status window shows the status of some elements of the bench and test equipment.

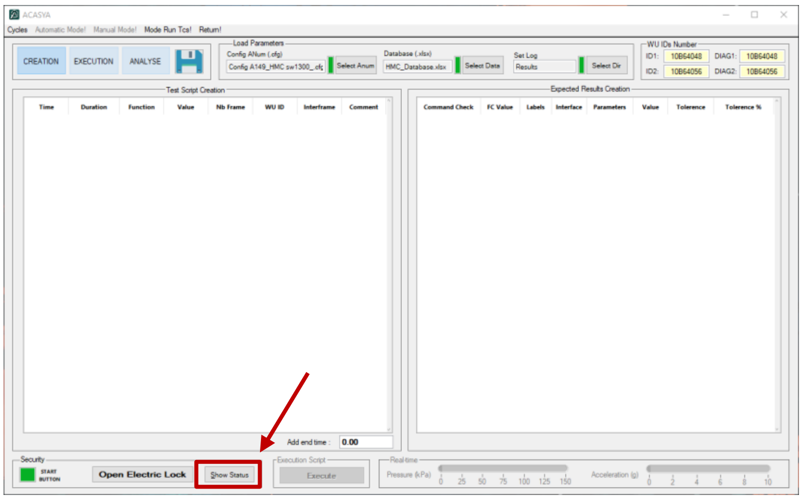
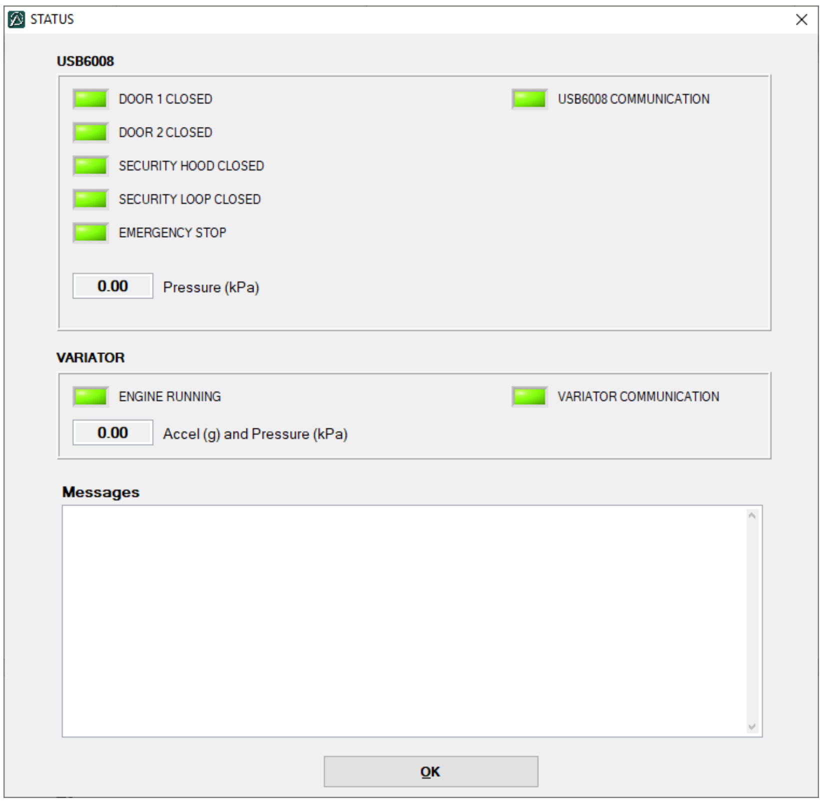


Figure 21. Status button position

This monitoring window allows you to check the status of doors, security hood, security loop, emergency stop, USB6008 communication, etc.


STATUS

USB6008

🟩 DOOR 1 CLOSED
🟩 DOOR 2 CLOSED
🟩 SECURITY HOOD CLOSED
🟩 SECURITY LOOP CLOSED
🟩 EMERGENCY STOP
🟩 USB6008 COMMUNICATION

0.00 Pressure (kPa)

VARIATOR

🟩 ENGINE RUNNING
🟩 VARIATOR COMMUNICATION

0.00 Accel (g) and Pressure (kPa)

Messages

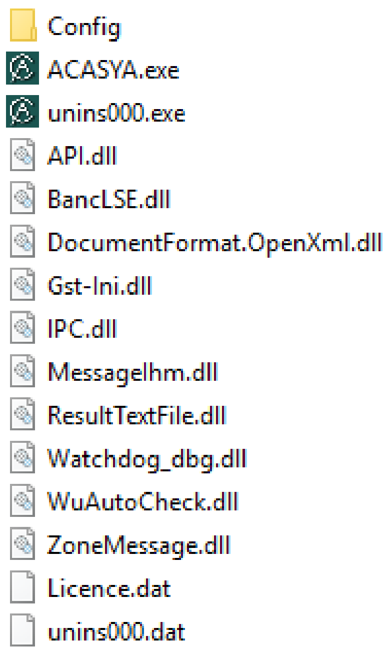
OK

Figure 22. Status window

APPENDIX

VI. APPENDIX

A. Installation files



Config
ACASYA.exe
unins000.exe
API.dll
BancLSE.dll
DocumentFormat.OpenXml.dll
Gst-Ini.dll
IPC.dll
Messagelhm.dll
ResultTextFile.dll
Watchdog_dbg.dll
WuAutoCheck.dll
ZoneMessage.dll
Licence.dat
unins000.dat

Figure 23. Installation files

B. Check Results example

1. PASSED example

```
c:\Users\uic80920\Desktop\WUTT\Seq_10-09-20_11-29-04\TestPM_AccFinale_t 10-09-
2020 11-29-37\Logs.xml

Check Report 10-9-2020


||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
|| TEST FOR WU ID "10B64048"
||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||


***************CheckCompareAcc***************
                 Check_1

Between in (1000 ms) and out (30000 ms)
Tolerance value: 10.000000
Tolerance [%]: 0.000000


================================================
|                CheckCompareAcc               |
|                                              |
| Valid Acceleration                         8 |
| Invalid Acceleration                       0 |
|                                              |
|                PASSED   8/  8                |
================================================


||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
|| TEST FOR WU ID "10B64056"
||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||


***************CheckCompareAcc***************
                 Check_1

Between in (1000 ms) and out (30000 ms)
Tolerance value: 10.000000
Tolerance [%]: 0.000000


================================================
|                CheckCompareAcc               |
|                                              |
| Valid Acceleration                         8 |
| Invalid Acceleration                       0 |
|                                              |
|                PASSED   8/  8                |
================================================
```

2. FAILED example

```
c:\Users\uic80920\Desktop\WUTT\Seq_10-09-20_11-29-04\TestPM_PressOnly2_t 10-09-
2020 11-39-18\Logs.xml

Check Report 10-9-2020


|||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
|| TEST FOR WU ID "10B64048"
|||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||


***************CheckCompareP***************
                 Check_1

Between in (1000 ms) and out (29000 ms)
Tolerance value: 10.000000
Tolerance [%]: 0.000000

The pressure 100.000000 is not between [101.890974,121.890974], (frames nb 14)

================================================
|                CheckCompareP                 |
|                                              |
| Valid Pressure                             7 |
| Invalid Pressure                           1 |
|                                              |
|                FAILED   1/   8               |
================================================


|||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
|| TEST FOR WU ID "10B64056"
|||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||

***************CheckCompareP***************
                 Check_1

Between in (1000 ms) and out (29000 ms)
Tolerance value: 10.000000
Tolerance [%]: 0.000000


================================================
|                CheckCompareP                 |
|                                              |
| Valid Pressure                             4 |
| Invalid Pressure                           0 |
|                                              |
|                PASSED   4/   4               |
================================================
```

ACASYA for WFC

- User Manual -

Author: Maxime PAGES

September 2020

Continental