Pulse program editor and compiler UI

### Contents

[Contents 1](#_Toc185260993)

[1 The pulse program editor and compiler menu 3](#_Toc185260994)

[1.1 The File menu 3](#_Toc185260995)

[Open pulse program 3](#_Toc185260996)

[Save pulse program 3](#_Toc185260997)

[New pulse program from existing file 3](#_Toc185260998)

[Rename current pulse program 3](#_Toc185260999)

[Open folder containing pulse program files 4](#_Toc185261000)

[Exit 4](#_Toc185261001)

[1.2 The edit menu: 4](#_Toc185261002)

[Undo: 4](#_Toc185261003)

[Cut: 4](#_Toc185261004)

[Copy: 4](#_Toc185261005)

[Paste: 4](#_Toc185261006)

[Select all: 4](#_Toc185261007)

[Indent: 4](#_Toc185261008)

[Unindent: 4](#_Toc185261009)

[Block comment: 4](#_Toc185261010)

[Block uncomment: 4](#_Toc185261011)

[Sort selection: 4](#_Toc185261012)

[Find down: 4](#_Toc185261013)

[Find up: 4](#_Toc185261014)

[Find and replace: 5](#_Toc185261015)

[Find and replace globally: 5](#_Toc185261016)

[1.3 The View menu: 5](#_Toc185261017)

[Go to line number: 5](#_Toc185261018)

[Go to procedure: 5](#_Toc185261019)

[Go back: 6](#_Toc185261020)

[Go forward: 6](#_Toc185261021)

[Show procedure syntax: 6](#_Toc185261022)

[Show macro procedures: 6](#_Toc185261023)

[Increment font size: 6](#_Toc185261024)

[Decrement font size: 6](#_Toc185261025)

[Show fault line: 6](#_Toc185261026)

[Command help 6](#_Toc185261027)

[1.4 The Compile menu: 6](#_Toc185261028)

[Compile current pulse program: 6](#_Toc185261029)

[Compile all pulse programs in current menu: 6](#_Toc185261030)

[Compile all pulse programs in menu list: 7](#_Toc185261031)

[1.5 The Procedure menu: 7](#_Toc185261032)

[1.6 The Help menu: 7](#_Toc185261033)

[1.7 Control buttons 7](#_Toc185261034)

[Compile: 7](#_Toc185261035)

[Help: 7](#_Toc185261036)

[Close: 7](#_Toc185261037)

[2 File information 7](#_Toc185261038)

[2.1 The text editor tab interface 8](#_Toc185261039)

[Pulse program 8](#_Toc185261040)

[Experiment control 9](#_Toc185261041)

[Default parameters 10](#_Toc185261042)

[User interface 10](#_Toc185261043)

[Importer file 10](#_Toc185261044)

[2.2 The status bar 11](#_Toc185261045)

This document describes the functionality of the menus and other controls in the pulse program editor and compiler. This can be displayed for any experiment by holding down the shift key when selecting the experiment from the main menu. Here is a typical user interface (in this case for the HSQCtgME experiment)

A screenshot of a computer program

Description automatically generated

Here is a description of the main UI features:

At the top we have a number of menu items. Many of these are related to the editors, but some are specific to the pulse program environment. Here is a description of each menu item:

# The pulse program editor and compiler menu

## The File menu

### Open pulse program

Generally, you open a pulse program using the main menu short cuts, however you can also load a pulse program using this interface. Just select the folder containing the pulse program files and they will be loaded into the editor.

### Save pulse program

This saves any changes made in the pulse program files. Unsaved changes are indicated by an asterisk after the affected filename.

### New pulse program from existing file

Generating a new pulse program from scratch is not recommended, the files are generally too complex for this. Rather use this option to copy an existing pulse program and change all the internal names to the selected new one. The first window to appear when choosing this option is a folder browser. Here you select the pulse program folder to be used as a template – by default it will select the currently loaded experiment. Next a small window will appear where you can define the location of the new experiment and its name:

A screenshot of a computer program

Description automatically generated

The dropdown menu in the path allows selection of existing pulse program folders or you can add a new one.

Once the new file name has been chosen, press the OK button to generate the necessary pulse program files. Note that in addition to copying the files to the new name it also includes changing the experiment name wherever it appears in the various files.

### Rename current pulse program

This option can be used to rename a pulse program. It will change the names of all relevant files as well as internal references to this file. Note that experimental data which was previously collected using this experiment will no longer be loadable. In this case you will need to use the option ‘Rename experiment folders’ in the main interface File menu.

### Open folder containing pulse program files

This just opens a Windows file explorer dialog to show the various files in the pulse program folder.

### Exit

Closes the pulse program editor and compiler. It will prompt you if files need to be saved.

## The edit menu:

### Undo:

Undoes the last action performed in the editor. Note that global find and replace actions are not included so be sure to make a backup first if the file is important!

### Cut:

Remove the selected text and copy to the clipboard.

### Copy:

Copy the selected text to the clipboard.

### Paste:

Replace the selected text with the contents of the text clipboard.

### Select all:

Select all the text in the current file.

### Indent:

Indent the selected text by 3 spaces (note, tabs are not used in the Prospa editors)

### Unindent:

Remove 3 spaces from the front of the selected text.

### Block comment:

Add a comment symbols # to the front of the selected text.

### Block uncomment:

Remove a comment symbol # from the front of the selected text.

### Sort selection:

Sort the selected lines alphabetically.

### Find down:

Search downwards in the file for the next instance of the selected text.

### Find up:

Search upwards in the file for the next instance of the selected text.

### Find and replace:

Open a file and replace dialog. This allows text to be replaced below the current insertion point. Note that global replacements are disabled. *Ignore case* and *Search up* are self-explanatory while *Wrap* will search from the top again if the end of the file is reached (or from the bottom if *Search up* is selected).

A screenshot of a computer

Description automatically generated

### Find and replace globally:

This option allows the replacement of several strings with others. This is useful for changing from one nucleus to another in a particular experiment. Each old and new string should be delimited with a comma. This means that you can’t replace strings which include commas.

A screenshot of a computer

Description automatically generated

Note that this replacement is applied to all pulse program files and there is no undo option, so don’t do this unless you have a backup of the folder.

## The View menu:

### Go to line number:

Select a line number and the editor will scroll to make that line visible and highlight the line. Note this is 1 based.

### Go to procedure:

If there is a procedure call in the text you are viewing you can place the insertion point anywhere on the procedure or macro name (if present) and this option will load the procedure into the editor along with the rest of the associated macro. Use the *Go back* option to return to the calling program. Use control double-click as a short-cut.

### Go back:

This returns from the current procedure to the calling program if the Go to procedure item was used to get there. Use F5 as a short cut.

### Go forward:

If you have used the Go back command to return to a procedure call, this option will undo this step. Use F6 as a short cut.

### Show procedure syntax:

By default, built-in Prospa and DLL commands will show their syntax in the editor status bar (at the bottom of the editor) if you type them in or click on them with the mouse. You can do the same for Prospa procedures without visiting the procedure file by shift-clicking with the mouse with the insertion point on the procedure name. Note that the command will not work from the menu – it is simply for documentation purposes.

### Show macro procedures:

With the insertion point on a macro or class name this will pop up a window with a list of all procedures in that file/class. You can then select from this list and the name will be copied into the editor.

### Increment font size:

This will increase the current editor font size by 1 point. Maximum size is 18 points.

### Decrement font size:

This will decrease the current editor font size by 1 point. Minimum size is 6 points.

### Show fault line:

If an error was detected when running a macro, then with this option you can jump to the line which caused the error. Be aware that if the error line was part of a try-catch structure then the error line will possibly be incorrect, and it will simply show the catch location. This is a very common problem in the Expert environment.

Command help***:***

If a Prospa or DLL command is selected, then this option will display help for this command in the help viewer if the relevant help file is present.

## The Compile menu:

### Compile current pulse program:

This will compile the currently loaded pulse program, updating several automatically generated procedures. This is the same as pressing the ‘Run’ button.

### Compile all pulse programs in current menu:

This will compile the all the pulse programs in the current parent directory (i.e. the menu where this pulse program appears).

### Compile all pulse programs in menu list:

This will compile all the pulse programs in all the user menus which are currently visible. This is useful if a significant update is made to the software, and you are advised to update all the pulse programs. (As some may not be included in the update). It is advisable in this case to make a backup of any critical experiments before doing this.

## The Procedure menu:

This menu lists all the procedures in the current file. Selecting one of these will cause the editor to scroll to the procedure and make it visible.

## The Help menu:

Here there is an option to display this file and also the pulse programming guide.

## Control buttons

A screenshot of a phone

Description automatically generated

### Compile:

This will compile the currently loaded pulse program updating several automatically generated procedures.

### Help:

This option will display this file

### Close:

Closes the pulse program editor and compiler. It will prompt you if files need to be saved.

# File information

A screenshot of a computer

Description automatically generated

In this group of controls, we have the *Output directory*. This is the directory where the experiment folder is stored. This folder contains several files, some of which are displayed in the text-editors below this group.

Next, we have *Pulse-program base name*. This is the base or core name of the experiment. In this case HSQCtgME. To this we add additional suffixes to define the various file types.

*Lock mode*. This is a rarely used feature which should only be selected if compiling an experiment for the lock channel. It is expected that this will only be used by Magritek staff.

*U.I. version*. This allows the generation of different versions of the files. The current version is 5. If you need to generate a version for an older version of Expert you have the choice of 3 and 4. However it is not expected that this will be needed very often. Note that the UI version number is saved to the Experiment control macro in the title comment. This should not be removed. Also note that these older versions will also require significant manual modification to the files before they will work.



## The text editor tab interface



Here you have the possibility to select and edit a number of different files. What is displayed here will depend on whether you have a DSP or FX3 spectrometer and whether you have selected the *Allow importer editing* option in the Preferences dialog General page:



With the importer editing option selected the tabs are extended by one.



Here is a short description of each file:

### Pulse program

This is the file with the name baseName\_pp.mac (e.g. HSQC\_pp.mac). This contains the *pulse\_program* procedure which includes a description of the UI parameters to display, the pulse program itself, and a relationships list connecting the UI parameters and the pulse program parameters. Recompile the experiment if any changes are made to this procedure.

In addition, the pulse program file will also include a procedure called *getFactoryBasedParameters* which returns a sublist of parameters appropriate for your spectrometer such as pulse length and amplitudes. For some more complex sequences there may be additional procedures.

### Experiment control

This file has the name baseName.mac (e.g. HSQC.mac). This contains entry points to the experiment – either as selected from the main menu or when executed from the ‘Run’ button. Here are the typical procedures found in this file:

*baseName* (replace with actual name): This will add the experiment to the parameter list and update the graph layout. If the shift key is held down, it will display the pulse program editor with this experiment loaded. This procedure is automatically generated when compiling the experiment.

*backdoor*: this is the entry point when the experiment is called from the ‘Run’ button or the ‘RunExpt’ script command. This executes the experiment. This procedure is automatically generated when compiling the experiment.

*getseqpar*: this is a summary of the important pulse program parameters and means that at least in the case of DSP spectrometers that the pulse program file need not be accessed when running an experiment. This procedure is automatically generated when the experiment is compiled. It is executed when the backdoor procedure is called.

*execpp*: execute pulse program. This is a user generated procedure which controls the experiment. It is called indirectly from the backdoor procedure. This contains commands to set various parameters on the spectrometer, run the sequence and return the data. It also typically processes and displays the data as well. Loops control the number of scans and any parameters which must be varied during the experiment. Finally, it will call procedures to save the collected data.

*getplotinfo*: this is a user generated procedure which contains a list connected the plot names with associated files. This is used when saving the data in execpp and when reloading data when an item in the history list is selected.

*expectedDuration*: this is an optional user generated procedure which define show long the experiment will take. For repTime based experiments this is simply the number of scans multiplied by the repTime, while for ieTime based experiments it is necessary to do a more detailed calculation of the experiment time by calling the pulse program with the necessary parameters and summing up various delays. This procedure is required to allow the progress bar to work.

*saveProcPar*: an optional user generated procedure which allows collected data to be reprocessed using Expert. It generates a file called proc.par which contains the necessary parameters for the 1D and 2D FT dialog to work.

*helpInfo*: an optional user generated procedure which specifies the location of an html help file in the documentation folder.

Other support procedures may also be present.

### Default parameters

This file has the name baseNameDefault.par (e.g. HSQCtgME.par). This file contains a list of default parameters for the experiment. When an experiment is selected from the main menu the parameter list will first be populated by default values before calling the *getFactoryBasedParameters* procedure in the pulse program macro. Parameters are defined using the syntax:

variableName = value

Comments and empty lines can be included to improve readability. It is recommended that parameters are sorted according to their parameter group in the parameter UI. Within these groups they can be sorted alphabetically.

### User interface

This stores separate procedures to define the parameter UI, plot run and load layouts and post-processing controls.

*interfaceDescription:* This contains a compact version of the parameter UI as defined in the pulse program. This procedure is automatically generated when compiling the experiment.

*plot\_run\_layout:* this returns the plot layout which should be displayed when running an experiment. This information is typically accessed from the Experiment control macro procedure execpp using the ucsPlot:getPlotReferences procedure call. This is a required, user defined procedure.

*plot\_load\_layout:* When an experiment is selected from the history list it will use this procedure to define the plot layout. You can also define the type of file here as ‘fid’ or ‘spectrum’. This just causes a warning to be issued if you try and overwrite raw fid data. This is a required, user defined procedure.

*processing\_controls*: This defines a group of user-defined buttons which will appear on the lower right side of the main window when an experiment has completed (or in some cases while it is running). These buttons can call different post processing procedure. The buttons can be attached to different plots, so they are only visible when the appropriate plot is selected.

### Importer file

This file contains procedures to important data collected by the Magritek standard software into Expert. This data set is typically missing some parameters needed by Expert as well as not including spectral information. It is therefore necessary to regenerate this missing information. Typically, this file will only be generated and edited by Magritek staff. Here are the important procedures. Currently only a few experiments support this option.

*addMissingParameters*: this adds the missing parameters using the defaults used in the standard software and the information stored in the existing acqu.par file.

*loadProcessAndDisplayData*: this loads the FID data and processes it to produce a spectrum. This information is then displayed.

Other procedures may be present to support the above primary procedures.

## The status bar



This is displayed at the bottom of the window. It has two sections; on the right is the current line number of the text insertion point. Note this is 1 rather than 0 based.

On the left is the syntax for the currently selected command. There are several options here:

A *general command* is one of the built-in Prospa commands (of which there are 517 at the time of writing this file).

A *DLL command* is a command defined in one of the DLLs found in the DLLs folder. These are also user definable if you have access to a C++ compiler like Visual Studio.

These commands will display syntax when simply clicking or selecting the command. However the following commands require control-clicking

A *procedure* command is a procedure written in the Prospa language.

A *class function* command is a command associated with a built-in class such as plot.

Argument types are indicated in a lighter grey (like int or cmat) while optional arguments are surrounded by square brackets.