WASPRUN Windows

Software by Douglas Instruments

Administrator Manual

Revision 4.1, October 2004
Compiled by Peter Baldock, Patrick Shaw Stewart and James Smith

Douglas Instruments Limited
Douglas House ♦ East Garston ♦ Hungerford ♦ Berkshire, RG17 7HD ♦ UK
www.douglasinstruments.com

CONTENTS

SCREENING WITH WASPRUN

General Description

Overview

WASPRun is a program to run screening experiments. It provides a user interface and supplies parameters to the WASP liquid handling script interpreter. The new Windows version of WASPRun provides a superset of the functionality of the original DOS version.

The purpose of WASPRun is twofold:

1. To provide a simple user interface, which clearly displays the main features of a crystallization experiment, and allows easy alteration of key parameters controlling the experiment.
2. To launch WASP, passing all parameters required by the experiment in an appropriately named Include file (.XPH).

Consequently, WASPRun is used in two viewing modes - User and Administrator modes. Administrator mode allows complete customization of all parameters supplied for a WASP .XPT experiment, whilst User mode allows restricted control of a subset of these parameters, most parameters usually being hidden from the user.

WASPRun provides two windows displaying current settings. The fundamental view is a “Group” window consisting of a number of group boxes containing related variables. A secondary view is the “PlateLoader” window, which gives a graphical representation of the layout of the experiment.

Groups

A Group is a collection of variables – usually related. There are currently three types of group:

1. User Defined Group – containing any arbitrary collection of variables, and constructed manually by adding variables as required. The name and caption of a User Defined Group can be set in Administrator mode.
2. Plate Specification Group – automatically generated, and containing the minimum number of variables necessary to specify a plate. Additional variables can be manually inserted into a Plate Specification Group, but this is not recommended.
3. Array Specification Group – automatically generated to provide the properties of a WASP array.

Groups may be added or edited only in Administrator mode, although the values of parameters within a group may be altered in User mode.

Variables

In Administrator mode, variables may be inserted in, edited or removed from an existing Group. Each variable has several properties:

- Name name of the parameter that will be passed to WASP.
- Caption legend associated with the text box displaying the variable’s value
- Type data type, picked from a dropdown list of possible types.
- Value text or numeric value of the variable. In the case of expressions, the Administrator will see the formula, whilst the user will see the calculated value.
- Hidden Boolean preventing display in User mode if set.
- Editable Boolean allowing changes to value in User mode if set.
- Suppress Output Boolean preventing passing of parameter to WASP if set.

When editing a list or enumerated variable, the list from which the value must be picked can be constructed manually, by adding new items in the list box provided. Highlighted items can be removed by pressing the adjacent [-] button. The automated groups create prefilled lists.

Visibility (Administrator/User)

All groups and all variables within each group are visible in Administrator mode. The visibility of a group in User mode can be controlled by the Hidden[] checkbox accessible in Administrator mode.

Each variable may be edited by double clicking the variable caption within the group, or by right clicking over the caption and selecting "Edit".Variables may be added by right clicking over an existing variable and selecting "Insert Before" or "Insert After". The resulting dialog allows the properties of the variable to be changed.

If the variable's Hidden bit is checked in the "Edit Variable" dialog, then the variable's caption will be displayed in red in its containing group when in Administrator mode. The variable will not be visible in its group in User mode.

If the Editable bit is checked then the user will be able to change the value of the variable. If the bit is clear, then the caption will be displayed grayed out, and whilst the administrator may change the variable value, the User will see a grayed out value, which cannot be changed.

Controlling .XPH Creation

WASPRun writes an include file with the .XPH extension, prior to launching WASP. The .XPH file contains a simple list of variable names and associated values. However not all variables are required by a typical WASP script; many are used for controlling the PlateLoader window, and are redundant as far as WASP is concerned.

Therefore, each variable has an associated "Suppress Output" bit. The administrator should check this bit for variables not likely to be needed by the .XPT file, although no problem is caused if all values are passed to WASP.

New .XPP Structure

The Windows version of WASPRun provides a superset of the functionality of the original DOS version. In order to achieve this, a new format for .XPP files has been defined. This is still a text based format, somewhat similar in layout to the original DOS format, but with fewer *ad hoc* characteristics, and with much greater flexibility.

Converting Dos format .XPP Files

Loading Old Format .XPP Files

WASPRun will seamlessly load a DOS format .XPP file. In the process, numeric variables which originally had associated lower and upper limits will be converted to simple numeric variables of the appropriate type, and additional warning expressions will be generated which correspond with the lower and upper limits.

Therefore, in place of each old variable, three new variables will be created. The attributes of the warning variables are automatically set to "Hidden=True" and "Suppress Output=True". It is recommended that the warning variable expressions are edited (given the greater power of the new expression evaluation) to more accurately reflect the experiment requirements. In so doing, several warnings can probably be removed.

Immediately an original file has been loaded, WASP can be run, with no change in functionality.

Saving Files in New Format

Selecting the menu item {File|Save} will overwrite the DOS file with the new format file. No backup file is produced. See Multiple File Conversion. It is therefore recommended that the original files are first copied to a backup directory, prior to being converted.

A file's format may be identified by the first line of text. This is currently "XScreen 1.1 Parameter File" for the latest release.

Multiple File Conversion

If, when the Open File Dialog is shown, multiple files are selected, then WASPRun will automatically load and save all the selected files. In the process, the file format will be updated to the latest version. WASPRun will create backup files, containing the original text, with the added extension ".bak".

At the end of this process, WASPRun currently displays the last file converted.

Adding Plates

Generating Plate Specification Groups

Plate specification Groups are automatically generated by selecting the menu item {Edit|Parameter Group List} followed by [New] in the ensuing dialog. Select [Plate Specification Group] and a new group will be created. The parameter group list will show the new group at the bottom of the list.

The information in the Plate Specification Group is used by the PlateLoader window, and consequently the name of the group is indexed ([n]), as is the caption. You may alter the caption, but if you alter the name, then the information coming from the variables in the group cannot be used by the PlateLoader display.

Variables within a Plate Specification Group

There are currently only six variables automatically created in a Plate Specification Group. They are not hidden, but not user editable, by default. These are :

1.	PlateType[n]	(List)	List of all plates available in Plates(…).dat
2.	PlateAtX[n]	(Float)	Offset of plate in X direction
3.	PlateAtY[n]	(Float)	Offset of plate in Y direction
4.	PlateAtZ[n]	(Float)	Offset of plate in vertical direction
5.	PlateIsDestination[n]	(Boolean)	Controls display in PlateLoader view.
6.	PlateSelectable[n]	(Boolean)	User can select plate

If the plate is aligned automatically in the .XPT file, then the offset entries should be removed.

The PlateType[] list should be reduced to contain only plates that the .XPT file can handle.

Having introduced a new Plate Specification Group, the Administrator should check the PlateLoader view before making alterations. It is generally recommended that all Plate Specification Groups be hidden. However, if it is intended that the user should be able to select the plate, then just the PlateType variable should be exposed.

Remember that captions for variables and groups can be changed by the Administrator to enhance clarity, without affecting the functionality of the linkage with WASP.

Adding Arrays

Generating Array Specification Groups

Array Specification Groups are automatically generated by selecting the menu item {Edit|Parameter Group List} followed by [New] in the ensuing dialog. Select [Array Specification Group] and a new group will be created. The parameter group list will show the new group at the bottom of the list.

The information in the Array Specification Group is used by the PlateLoader window, and consequently the name of the group is indexed ([n]), as is the caption. You may alter the caption, but if you alter the name, then the information coming from the variables in the group cannot be used by the PlateLoader display.

Variables within an Array Specification Group

There are currently fourteen variables automatically created in an Array Specification Group. Some are hidden, but none are user editable, by default. These are :

1.	ArrayName[n]	(Text)	Name of array as found in .XPT file.
2.	ArrayPlate[n]	(Integer)	Plate on which this array is defined.
3.	ArrayStartX[n]	(Integer)	Array left hand column.
4.	ArrayStartY[n]	(Integer)	Array top row.
5.	ArrayWidth[n]	(Integer)	Number of columns in array.
6.	ArrayHeight[n]	(Integer)	Number of rows in array.
7.	ArraySelectable[n]	(Boolean)	Checked if user can define arrays.
8.	ArrayFillOrder[n]	(Enumerated)	One of eight possible filling orders.
9.	ArrayCount1[n]	(Integer)	Number of wells filled with first color.
10.	ArrayFillText1[n]	(Text)	Text used to annotate first set of wells.
11.	ArrayColor1[n]	(List)	Fill color for first set of wells.
12.	ArrayCount2[n]	(Integer)	Number of wells filled with second color.
13.	ArrayFillText2[n]	(Text)	Text used to annotate second set of wells.
14.	ArrayColor2[n]	(List)	Fill color for second set of wells.

Customizing Array Specifications

Several of these items may be redundant to obtain a correct display in the PlateLoader window, but they are automatically generated to avoid the need for special knowledge in the naming of the variables. It is not advisable to change the variable names, as the PlateLoader display will then not function properly. However, the captions may be altered to improve clarity.

The values should be set to correspond with the logic of the associated .XPT file. In particular, the array name should match that used in the .XPT file, and the plate number to which the array applies should be entered.
The geometric values (StartX, StartY, Width and Height) can be hardwired as constants, or the Selectable property can be checked, allowing the user to drag out their own definition of the array in the PlateLoader window. Should the latter option be selected, then these four variables will be automatically updated with new values, when an array is redefined with the mouse on the associated plate.

Array Filling

Items 8 – 14 are provided purely for display, and control the visual content of the array. The PlateLoader window uses these values to give the user a realistic picture of the experiment layout. No problem will occur if these values are removed, but the resulting display will be less helpful.

An array may be filled in any of eight orders, defined by the starting corner and the direction of fill. It is important for the Administrator to select the correct order from the enumerated list, so that the PlateLoader display matches the behavior of the .XPT file, and the user is not misled into filling plates incorrectly. The eight orders are :

i. TopLeftHorizontal
ii. TopLeftVertical
iii. TopRightHorizontal
iv. TopRightVertical
v. BottomLeftHorizontal
vi. BottomLeftVertical
vii. BottomRightHorizontal
viii. BottomRightVertical

For generality, the PlateLoader window can show an array filled with two different categories of solution; for example, these might be viscous and non-viscous solutions. Associated with the two categories are two counts, two text constants to annotate the wells, and two fill colors. The annotation will be visible in wells with a large enough diameter to display correctly. If the sum of the two counts is less than the number of wells in the array, then the remaining wells are shown empty. If the sum exceeds the number of wells available, then the extra wells are ignored.

Dealing with Lists

Difference Between List Variables and Enumerated Variables

Lists variables and enumerated variables are useful mechanisms for constraining the user to selecting specific values There are two differences between a list variables and an enumerated variables:

- List variables export text to WASP whereas enumerated variables export an integer corresponding to the index of the item picked – first item = 0.
- List variables are sorted alphabetically, whereas enumerated variables maintain their entry order.

They are a little different to other variables in not only taking a value, but also requiring a list of possible values from which to pick. Several lists and enumerations are generated automatically, when Plate or Array Specification Groups are created. These variables have their lists of selectable items preprogrammed, or read from a global source such as Plates(…).dat. However, list variables introduced manually must have their lists set manually.

Inserting Items into a List

When the type of a variable is set to List or Enumerated, a dropdown list box labeled "List" is displayed in the "Parameter Details" editing dialog. Pull down the list box to see the currently available items – see Copying Lists. To add a new list item, enter the item in the list text line and press <Enter>. The new item will be added to the list unless it is a duplicate, and remain selected on the list text line. This allows the next item to be entered by immediately overtyping, or a variant on the previous item to be entered by editing the previous text.

To remove an item from the list, pull down the list and select the offending item. Then press the adjacent [-] button. Successive presses of the [-] button will remove successively earlier items, until the first item has been removed. Then successively later items will be removed until the list is empty.

Copying Lists

As an aid to creation of new lists, the "Parameter Details" dialog remembers the last list that was edited. If you are creating a new list that is similar to an existing list, first open the "Parameter Details" dialog for the existing list variable, and then immediately close it. Now create the new variable – when you set the data type to List or Enumerated, the list will already contain the items from the existing list variable. This only applies to the creation of new lists; obviously, when editing an existing list, the dialog will reload the items for that list.

Creating Expressions to Couple Variables

The Need for Expressions

WASPRun uses the principle of high redundancy in specifying parameters, both for export to WASP in executing an experiment, and in its mechanism for displaying the experiment layout in the PlateLoader window. Every automatically generated group defines a significant number of variables to cater for most conceivable configurations.

Sometimes, as a matter of simplification, when these variables are not required they can simply be removed. However, it is frequently the case that a variable in one group should be related in some way to a variable in another group. Examples of this are

- There are constraints on variables which depend on the values of other variables. For example, the experiment might demand that the shape of one array is the same as another.
- It may be useful to display calculated information. For example, the user might like to see the total volume of protein used, having specified the number of wells and the size of a drop.
- Variable mapping may be required to match the new WASPRun presentation with existing .XPT files. Older files may use arbitrary variable names such as dStartRow and dStartCol for the position of the destination array, whereas WASPRun automatically generates Array Specification Groups with indexed variables ArrayX[n] and ArrayY[n]. In this case, expressions allow the Administrator to map automatically generated variables onto variables required by the experiment.

See Expressions and Warnings

Providing Warnings

Mechanism

The Warning data type is really an expression which, if it evaluates to a number other than zero, triggers a special action. Warnings expressions typically use the comparison operators (==, !=, >, >=, <, <=). When the warning expression evaluates to a non-zero value, a warning message is issued, quoting the caption of the warning variable. This caption can be edited by the Administrator, to make clear the condition which triggered the warning. Warning variables are generally hidden from the user.

Automatic Generation

When WASPRun loads an original DOS .XPP file, each numeric variable – which originally had associated minimum and maximum limits – will give rise to two warning expressions. Given the greater power of the current WASPRun, many of these automatically generated warnings may be removed and replaced with a smaller number of more intelligent warning expressions.

Data Types – Summary

Available Data Types

Each variable in a group must be assigned a data type. The possible data types are :

• Text	String. The field width (maximum number of characters) should be specified.
• Integer	16 bit signed number.
• Float	Single precision floating point value. The precision should be specified, and denotes the number of decimal places.
• Boolean	Logical true/false represented by a checkbox.
• List	Text item selected from a list of text items.
• Enumerated	Integer representing index of text item in a list
• Expression	Algebraic expression in terms of other numeric variables
• Warning	Expression evaluating to true if not equal to zero.

Numeric Data Types

Numeric data types take two forms:

• Integer	16 bit signed integer
• Float (real)	32 bit single precision IEEE standard Floating Point

Automatic conversion takes place between types; if expressions use both integers and reals, integers are promoted to reals as the evaluation takes place.

In defining a variable, the Administrator should specify the precision of the variable. With reals, the precision specifies the number of decimal places used in display and output, but does not change the precision of calculation. With integers, the precision specifies the field width of the formatted output – that is to say, the amount of space padding prior to the digits of the number; this is usually unimportant.

Text Data Types

Text variables are the simplest. Whatever characters are entered into the variable's text box are treated as the value of the variable. The precision associated with a text variable refers to the number of characters displayed before truncation. The precision should normally be chosen not to interfere with likely values for the variable.

Lists

Lists and enumerated types prevent the user from inadvertently selecting illegal or unpredicted values. In other respects they behave as text variables and integer variables respectively. See Dealing with Lists.

Expressions and Warnings

Syntax of an Algebraic Expression

Currently WASPRun supports only algebraic expressions. An expression is similar to the right hand part of a computer program statement. Examples of expressions are:

```
7/5
ArrayPlate[platenum]
(nTotal-nViscous)*vDrop
```

The variable containing the expression is implicit – it “owns” the text box in which the expression resides. When defining the expression for variable x you enter “nTotal/100” rather than “x = nTotal/100”.

Expressions obey normal algebraic precedence. See Operators.

Addressing Numeric Arrays

WASPRun currently supports indexed numeric 1-D arrays. Multidimensional arrays are not supported. Note that square brackets [] are used in C/C++ and Pascal style. An array is internally indexed as a collection; the index can therefore be either integer or float, but attempting to access an array value that has not been defined causes an error.

Operators

Arithmetic

+	Integer or real addition	Precedence = 1
-	Integer or real subtraction	
*	Multiplication	Precedence=2
/	Division	
%	Modulus (remainder after division)	
**	Raise to power (including non integer powers)	Precedence=3
(...)	Evaluate contents of brackets	Precedence=4
[...]	Array Index by contents of Brackets	Precedence=5

Bit

&	Bitwise AND	Precedence = 1
|	Bitwise OR	
!	Bitwise NOT	
^	Bitwise XOR	

Comparison

= =	Equal to	Precedence=0
!=	Not Equal To	
>	Greater Than	
>=	Greater Than or Equal To	
<	Less Than	
<=	Less Than or Equal To	

Logical

&&	Logical AND	Precedence=0
||	Logical OR	

Functions

The usual mathematical functions are provided. These are:

• Sqr(*value*)	Returns the square root of *value*
• Exp(*value*)	Returns e raised to the power of *value* (e***value*)
• Ln(*value*)	Returns the natural logarithm of *value*
• Sin(*value*)	Returns the sine of *value* radians
• Cos(*value*)	Returns the cosine of *value* radians
• Tan(*value*)	Returns the tangent of *value* radians
• Atan(*value*)	Returns the angle, the tangent of which is *value*
• Int(*value*)	Returns the integer part of *value* (rounds down)
• Round(*value*, n)	Returns the *value* rounded to n decimal places.
• Min(*value1*, *value2*)	Returns the minimum of *value1* and *value2*
• Max(*value1*, *value2*)	Returns the maximum of *value1* and *value2*

Known Bugs and Shortcomings

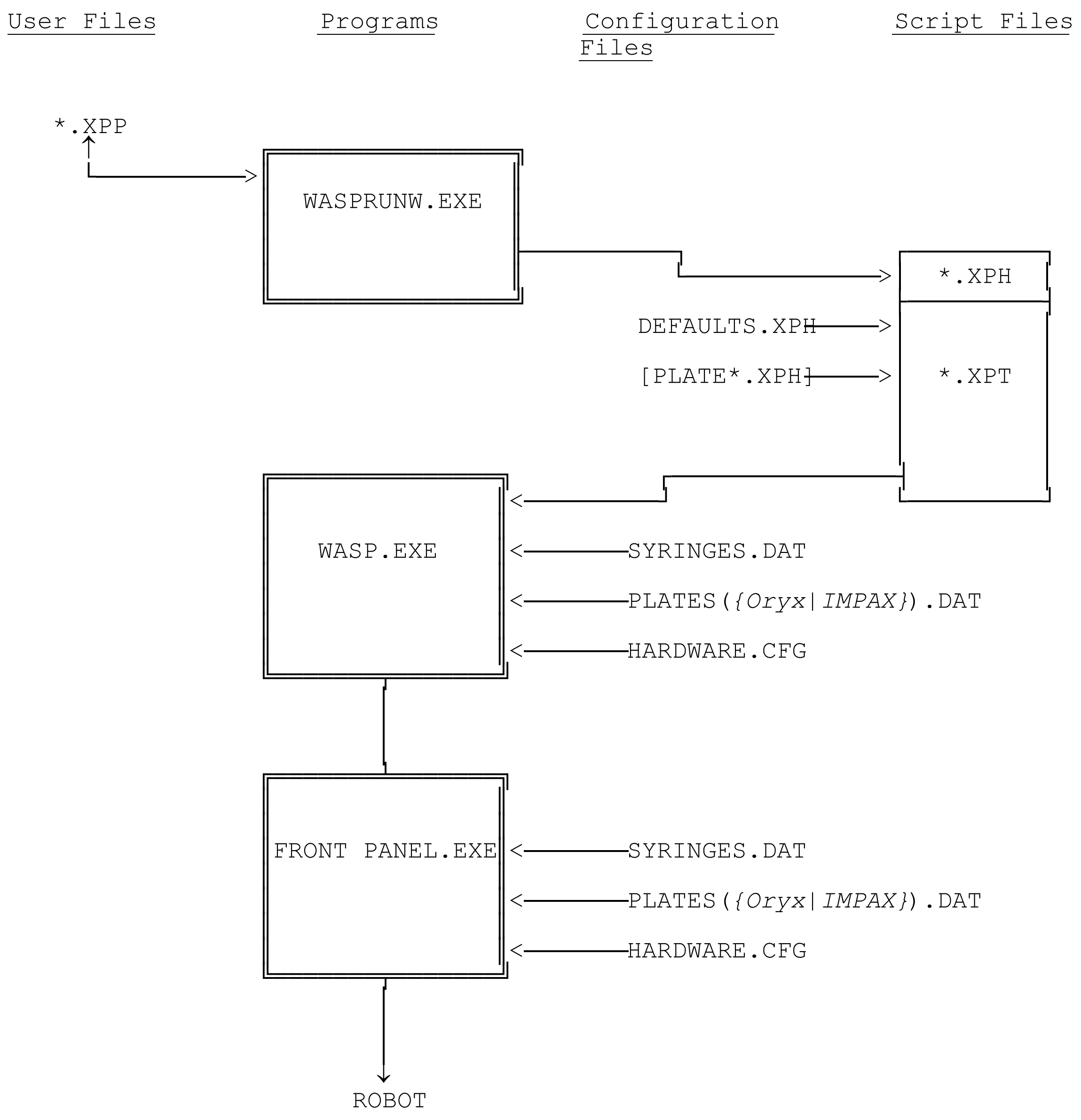
Bugs

- Occasionally, changing a variable's name may cause an error. This is an obscure bug and we are working on it. Save your file before editing variable names.
- Expressions involving the raise-to-power operation "**" do not always continue evaluation after the ** operator.

Shortcomings

- Random information(such as solution contents for each well) cannot be displayed.
- There is currently no means of specifying well-by-well properties for an experiment, although WASPRun can pass the names of Include files to WASP which contain arrays of well property values; this is the recommended method.

File Relationships for WASPRUN


User Files
Programs
Configuration Files
Script Files
*.XPP
WASPRUNW.EXE
*.XPH
DEFAULTS.XPH
[PLATE*.XPH]
*.XPT
WASP.EXE
SYRINGES.DAT
PLATES({Oryx|IMPAX}).DAT
HARDWARE.CFG
FRONT PANEL.EXE
SYRINGES.DAT
PLATES({Oryx|IMPAX}).DAT
HARDWARE.CFG
ROBOT