

# User Guide

## **Power Assist™**

**Version 4.01**

**August 1995**

**Lucas Control Systems Products  
Deco™ Systems**

Information in this document is subject to change without notice. Companies, names and data used in examples herein are fictitious unless otherwise noted. No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of Lucas Automation and Control, Inc., Lucas Control Systems Products, Deeco Systems.

© 1995 Lucas Automation and Control, Inc. (LACE) All rights reserved.

Deeco Systems is a trademark of LACE, all other trademarks are the property of their respective owners.

Document Part No. 12846  
Printed in the United States of America

---

# TABLE OF CONTENTS

<b>INTRODUCTION.....</b>	<b>1</b>
<b>INSTALLATION.....</b>	<b>2</b>
MOUSE.....	4
I/O DRIVERS.....	4
<b>TUTORIAL.....</b>	<b>5</b>
<b>DEVELOPING AN APPLICATION.....</b>	<b>18</b>
CREATING DRAWINGS .....	20
<i>Paintbrush Tips</i> .....	22
USING POWER ASSIST DEVELOPMENT .....	24
<i>Tags</i> .....	24
Load .....	25
Save.....	25
Edit.....	26
Drivers Configuration Window .....	27
Blocks Window .....	29
Text Tags Window .....	30
Expressions Window .....	31
Groups of Tags Window .....	31
Mirror Files Window.....	32
Tags Window .....	33
<i>Insert</i> .....	44
Bitmap.....	45
Trend.....	46
Bargraph.....	48
Table .....	49
Text.....	50
Setpoints.....	52
Fill.....	54
Keys .....	55
Display .....	55
Mouse.....	56
Meter.....	58
Monitor.....	59
<i>Edit</i> .....	60
Modify .....	60
Delete.....	60
Copy .....	60
Move .....	60
Clear.....	60

Monitor .....	60
<i>Screens</i> .....	<i>61</i>
Load .....	61
Save .....	61
Cycle.....	61
Information.....	62
Monitor .....	62
<i>Settings</i> .....	<i>62</i>
Options .....	62
Monitor .....	62
<i>Application</i> .....	<i>63</i>
Load.....	63
Save .....	63
Edit.....	63
Functions Window .....	66
Histories Window .....	71
History Tags and Mirror Tags Windows.....	72
Alarms and Events Window .....	72
Procedures Window .....	73
Printing Colors Window.....	73
Run .....	73
<b>RUNNING THE APPLICATION .....</b>	<b>73</b>
SETPOINTS AND TEXT TAGS .....	75
ALARMS AND EVENTS .....	75
<i>Alarms Acknowledgment</i> .....	75
<i>Events</i> .....	76
ANALYZING DATA .....	76
<i>Historical Analysis</i> .....	77
<i>Formatted Numerical Historical Analysis</i> .....	78
<i>Statistical Process Control</i> .....	82
REPORTS .....	84
USING A NETWORK .....	84
SET TAG.....	85
PRINT.....	85
DEFAULTS .....	85
<b>APPENDIX A - POWER ASSIST FILES .....</b>	<b>87</b>
<b>APPENDIX B - I/O DRIVERS.....</b>	<b>87</b>
I/O DRIVERS AVAILABLE.....	89
BUILDING AN I/O DRIVER .....	89
<i>Parameters</i> .....	90
<i>Functions</i> .....	91

---

<i>Driver.h</i> .....	93
<i>Testing</i> .....	93
<b>APPENDIX C - SPC FORMULAS</b> .....	<b>94</b>
<b>APPENDIX D - OEMS</b> .....	<b>96</b>



---

# INTRODUCTION

Welcome to Power Assist, the simple, easy to use PLC operator interface from Lucas Control Systems Products, Deeco™ Systems. Power Assist's simple tools and intuitive interface allow you to create operator interface applications with minimal effort. Features include: drivers for most popular PLC families, 3000 tags, graphic importation utility, animation, alarm/event logging, historical analysis, statistical process control and recipes.

The *Power Assist User Guide* contains instructions for installation of the software, a tutorial and detailed information on the tools available for developing and running an application. Also included are sections on using the I/O drivers, building a custom I/O driver, SPC formulas and customizing the software for OEMs.

Although most questions can be answered by reading the appropriate section in the *Power Assist User Guide*, technical support is available from the factory 6:00 am to 5:00 pm Monday through Friday. For sales information and technical support please contact:

## **LCSP, Deeco™ Systems**

31047 Genstar Road

Hayward, California USA 94544-7831

Phone: 1-800-376-1154 or 510-471-4700

Fax: 510-489-3500

Asia/Pacific Fax: 510-471-3207

Technical Support BBS: 510-471-5402

Technical Support CompuServe Address: 76325,3043

**This page is blank.**

# INSTALLATION



The following table shows the minimum and the recommended configuration to run Power Assist:

Minimum Configuration	Recommended Configuration
IBM compatible with 386SX-25	IBM compatible with 486DX-25
Hard disk with 2 Mbytes free	Hard disk with 8 Mbytes free (with manual)
2 Mbytes of RAM	4 Mbytes of RAM
VGA mono	VGA color
DOS 5.0 or higher	DOS 5.0 or higher
1 serial port	2 serial ports
1 parallel port	1 parallel port
	Mouse
	<i>IBM Pro Printer</i> compatible printer
	Windows 3.1 <sup>1</sup>

A hard disk with a short access time improves the performance of the application especially when loading screens.



Power Assist is distributed on one 1.44MB floppy. Place the floppy in the A drive and type **A: <enter>**. Now type **INSTALL C: <enter>**. If you are installing Power Assist on a different drive, enter the corresponding drive letter.

<sup>1</sup>Windows is used just to create drawings while developing the application.

The *Power Assist User Guide* is also included on the distribution disk and is a Microsoft Word document. If you wish to load it onto your hard disk type **MANUAL** <enter>. The user guide requires approximately 5.5 Mbytes of disk space. Power Assist includes Power Assist Development and Power Assist Runtime. Power Assist Development, PA.EXE, is used to develop the application. Power Assist Runtime, PAX.EXE, is used to run the application.

## Mouse



Power Assist is compatible with the Microsoft Mouse and Deeco Systems DOS mouse driver LDTOUCH.COM

## I/O Drivers



I/O drivers are memory resident programs that are used to establish the communication between Power Assist and a specific data acquisition equipment. Before loading Power Assist you must install the I/O drivers that will be used by the application. Every I/O driver has its own DOC file that explains how to configure it. To install an I/O driver type **DRIVER FILENAME** <enter>. A message indicates the installation of the I/O driver. Consult the appropriate driver document for specific instructions for connecting to your PLC.



To install the driver:

`<driver> [/v n] ENTER`

`/v` - changes the interrupt used by Power Assist. Sometimes there is a conflict between the interrupts used by your PC and the one used by Power Assist (200). If Power Assist halts during its initialization there is a conflict. You can use this option to change the interrupt used by Power Assist.

`n` is the number of the interrupt (between 70 and 199).

## TUTORIAL

This section will lead you through creating an example application in Power Assist.

### 1. Creating the drawings used as screen background

The first step to create an application is to make the drawings used as background for the screens. Drawings must be saved in the Windows .BMP format.

In our example, we are going to use Paintbrush to generate the drawings. Before beginning the drawing you must check the option *Options-Image Attributes* in Paintbrush to be sure that the width is 640 and the height is 464 pixel.

Let us assume you have made two drawings, saved as DRAW1.BMP and DRAW2.BMP.

The next step is to proceed with the conversion of the drawings to the Power Assist VGC format. The VGC format is a compressed bitmap file used by Power Assist while running the application. The conversion is performed by the utility DIBVGC.EXE. The .BMP files must reside in the same directory as the utility.

To convert a file type:

```
dibvgc draw1.bmp <enter>
```

and

```
dibvgc draw2.bmp <enter>
```

or

```
dibvgc *.bmp <enter>
```

After the utility is run on draw1.bmp and draw2.bmp, two files will be produced; draw1.vgc and draw2.vgc. These files will be used as the background for your application.

## 2. Defining the tags

The next step is to load Power Assist Development and to define the tags used in the application. To load Power Assist Development type:

*pa* <enter>.

If you are using the demo version type:

*pademo* <enter>.

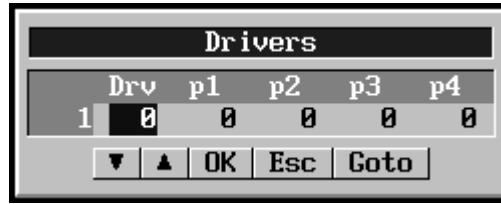
If Power Assist does not run or gives a memory error, check the interrupt as described in the installation section. If a memory error is given make sure that you have 600Kbytes of free conventional memory. Consult your DOS manual for information on how to free up memory.

Begin defining the tags by selecting *edit* in the *Tags* menu. The first window that will appear defines the number of drivers, blocks, tags etc.

TAGs configuration		
Number of drivers	:	<b>1</b>
Number of blocks	:	0
Number of text tags	:	2
Number of mirror files	:	0
Number of tags	:	30
Number of expressions	:	0
Number of groups	:	1
Number of retries	:	0

OK Esc

Let's use the default options in our example.  
Press <enter> or click OK to confirm the options  
and go to the driver(s) configuration window.



In our example we are going to use the demo driver, whose number is zero. The driver demo has no parameters, so press <enter> to confirm the default options. If you are using the full development package consult the driver document for information on the parameters.

The next window is used to define the text tags. Press <enter> to confirm the name of the default text tags. Press <enter> again to confirm the default options in Group of tags window.

The next window is used to define the tags:

Tags <TMP>											
	Name	Oper	Drv	n1	n2	n3	n4	Gp	?	Alfa	Beta
1	TAG0001	PLC	0	0	0	0	0	1	No	0.0000	0.0000
2	TAG0002	PLC	0	0	0	0	0	1	No	0.0000	0.0000
3	TAG0003	PLC	0	0	0	0	0	1	No	0.0000	0.0000
4	TAG0004	PLC	0	0	0	0	0	1	No	0.0000	0.0000
5	TAG0005	PLC	0	0	0	0	0	1	No	0.0000	0.0000
6	TAG0006	PLC	0	0	0	0	0	1	No	0.0000	0.0000
7	TAG0007	PLC	0	0	0	0	0	1	No	0.0000	0.0000
8	TAG0008	PLC	0	0	0	0	0	1	No	0.0000	0.0000
9	TAG0009	PLC	0	0	0	0	0	1	No	0.0000	0.0000

Let's define five tags as TEMP-1, TEMP-2, TEMP-3, TEMP-4 and TEMP-A. The first four will simulate four temperatures and the last one will contain the average value of these temperatures.

The operation (column Oper) of the first four tags will be *PLC*, which indicates that the values come from the I/O driver. The column driver will indicate the number of the I/O driver (0 for demo driver). The parameters n1 to n4 usually indicate the address of the variable in the data acquisition equipment. They are not used for the demo driver.

The fifth tag will have the operation average (*Aver*) and n1 and n2 will define the tags to be used in the average operation. The association to the tags is made by their index which appears on the left of the name of each tag. In this case, n1 will be 1 and n2 will be 4, indicating tags number 1 to number 4.

To enter the name of the tag and n1 to n4, position the cursor with the mouse or the arrow keys and type the desired value. The operation can be selected with the mouse, Space Bar, Ctrl →, Ctrl ← or typed.

Tags <TMP>											
	Name	Oper	Drv	n1	n2	n3	n4	Gp	?	Alfa	Beta
1	TEMP-1	PLC	0	0	0	0	0	1	No	0.0000	0.0000
2	TEMP-2	PLC	0	0	0	0	0	1	No	0.0000	0.0000
3	TEMP-3	PLC	0	0	0	0	0	1	No	0.0000	0.0000
4	TEMP-4	PLC	0	0	0	0	0	1	No	0.0000	0.0000
5	TEMP-A	Aver	0	1	4	0	0	1	No	0.0000	0.0000
6	TAG0006	PLC	0	0	0	0	0	1	No	0.0000	0.0000
7	TAG0007	PLC	0	0	0	0	0	1	No	0.0000	0.0000
8	TAG0008	PLC	0	0	0	0	0	1	No	0.0000	0.0000
9	TAG0009	PLC	0	0	0	0	0	1	No	0.0000	0.0000

▼ ▲ OK Esc Goto

Press <enter> to confirm the options.


Now select the option *Tags-Save* and save the tags as TEST.

### 3. Creating screens

Let's create the first screen by defining a background, one trend, one bargraph and one meter.

To define a background, select *bitmap* from the *insert* menu. With the mouse or the arrow keys and **<enter>**, select the desired drawing in the files window that appears on the left of the screen.

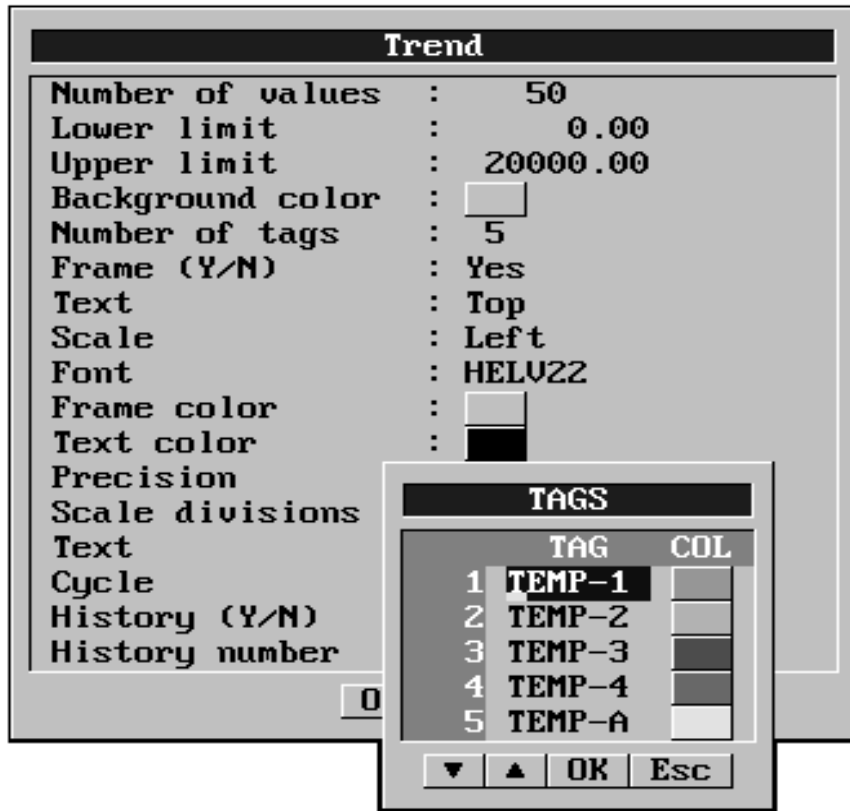
Now, select the option *Insert-Trend*. With the mouse, select an area for the trend.



The image shows a dialog box titled "Trend" with a list of configuration options. Each option is followed by a colon and its current value. At the bottom of the dialog are two buttons: "OK" and "Esc".

Option	Value
Number of values	50
Lower limit	0.00
Upper limit	20000.00
Background color	[Color Selection Box]
Number of tags	5
Frame (Y/N)	Yes
Text	Top
Scale	Left
Font	HELV22
Frame color	[Color Selection Box]
Text color	[Color Selection Box]
Precision	0
Scale divisions	10
Text	TEXT
Cycle	0
History (Y/N)	No
History number	1

Confirm the default options pressing **<enter>**.



Now you have to define the tags associated to this trend. You can change the tags with the mouse, Space Bar, Ctrl →, Ctrl ← or typing the new tag name. To change the colors, position the cursor over the color (with keyboard or mouse) and select the desired color in the color pad (with keyboard or mouse). In our example we will confirm the default tags and colors.

Now repeat the same procedure for the bargraph and the meter.

If you want to change, move, copy or delete any of the inserted objects, use the option *Edit*.

You can now save the screen. Select the option *Screens-Save* and save it as SCREEN1.

Select *Edit-Clear* to define the second screen. Define another screen and save it as SCREEN2.

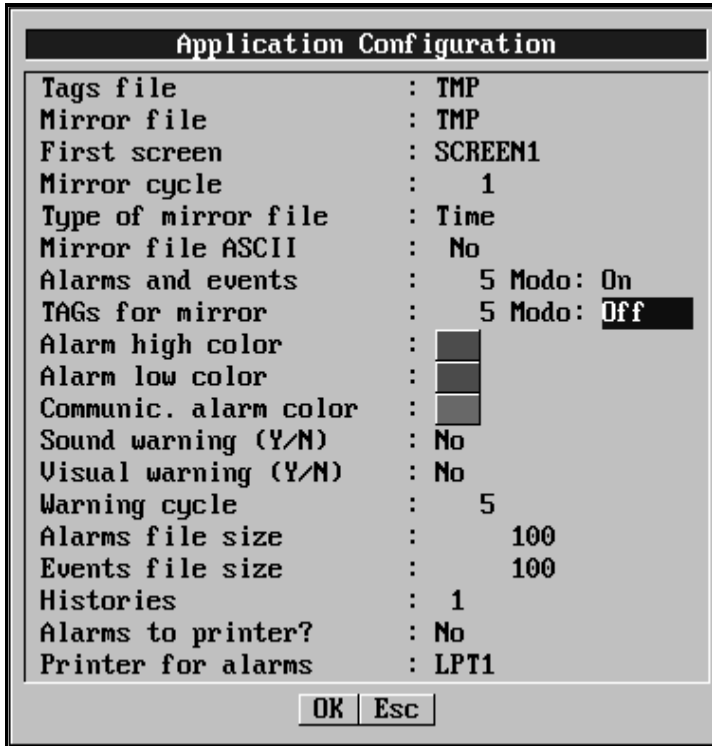
#### 4. Defining the application

To define the whole, select the option *Application-Edit*. The following screen will appear:

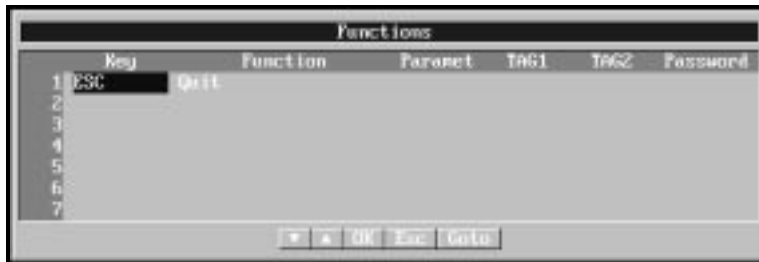
Application Configuration	
Tags file	: TMP
Mirror file	: TMP
First screen	: TMP
Mirror cycle	: 1
Type of mirror file	: Time
Mirror file ASCII	: No
Alarms and events	: 5 Modo: On
TAGs for mirror	: 5 Modo: On
Alarm high color	: <input type="checkbox"/>
Alarm low color	: <input type="checkbox"/>
Communic. alarm color	: <input type="checkbox"/>
Sound warning (Y/N)	: No
Visual warning (Y/N)	: No
Warning cycle	: 5
Alarms file size	: 100
Events file size	: 100
Histories	: 1
Alarms to printer?	: No
Printer for alarms	: LPT1

OK Esc

Define the first screen as SCREEN1 and set the Tags for mirror to *Off*.



Confirm the options with <enter> and go to the Functions window:



In this window, define F1 as *Change Screen* to SCREEN1, F2 as *Change Screen* to SCREEN2, F3 as *Historical Analysis* and F4 as *Acknowledge Alarms*.

Functions						
Key	Function	Paramet	TAG1	TAG2	Password	
1 ESC	Quit					
2 F1	Change screen	SCREEN1				
3 F2	Change screen	SCREEN2				
4 F3	Historical Analysis					
5 F4	Acknowledge Alarms					
6						
7						

▼ ▲ OK Esc Goto

Confirm the functions and go to the next screen to define the histories.

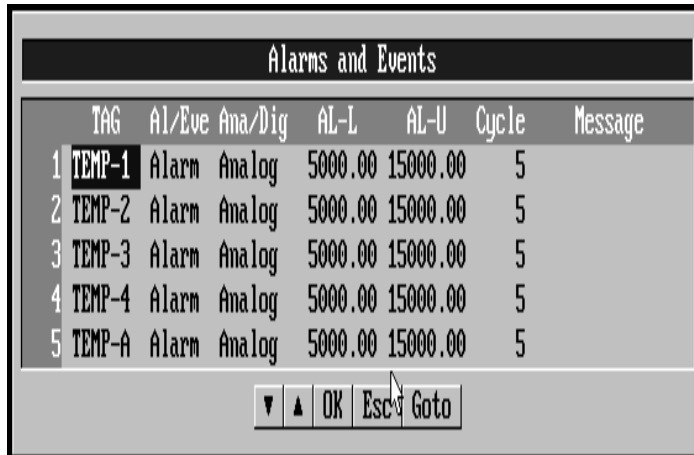
Histories						
Name	TAGs	Cycle	Process	Trigger	Size	
1 TMP1	5	2	Cont	Time	100	

▼ ▲ OK Esc Goto

Confirm the window default options and then go to the definition of the tags for the history:



Confirm the default tags and go to the definition of alarms:



Now define the alarm limits (*AL-Up* and *AL-Low*) messages according to the following example:

Alarms and Events							
	TAG	Al/Eve	Ana/Dig	AL-L	AL-U	Cycle	Message
1	TEMP-1	Alarm	Analog	400.00	19000.00	5	Temp Reactor 1
2	TEMP-2	Alarm	Analog	400.00	19000.00	5	Temp Reactor 2
3	TEMP-3	Alarm	Analog	400.00	19000.00	5	Temp Reactor 3
4	TEMP-4	Alarm	Analog	400.00	19000.00	5	Temp Reactor 4
5	TEMP-A	Alarm	Analog	400.00	19000.00	5	Average Temp

▼ ▲ OK Esc Goto

Confirm this window and the next two (Procedures and Printing colors).

Select the option *Application-Save* and save the application as TST.

## 5. Running the application

Now you can run the application by selecting the option *Application-Run*. Power Assist Development runs the application just for 10 minutes. To run the application for more than 10 minutes you have to use the Runtime (PAX.EXE):

Type: *pax tst* <enter>

Now that you are running the application, you can select the functions you have defined (F1, F2, F3 and F4) and see the result.

## DEVELOPING AN APPLICATION



You can develop an application very quickly using Power Assist. To load it type:

*pa* [/v *n*] <enter>

*/v* - changes the interrupt used by Power Assist. Sometimes there is a conflict between the interrupts used by your PC and the one used by Power Assist. If Power Assist halts during its initialization there is a conflict. You can use this option to change the interrupt used by Power Assist.

*n* is the number of the interrupt (between 70 and 200).

Using Power Assist is very easy. You can select the option in the menu using the keyboard or the mouse.



Using the keyboard you can select an option with the UP or DOWN arrow followed by ENTER or pressing the key corresponding to the capital letter of the option. The ESC key makes you return to the previous menu.



You can also select an option with the mouse by just pointing and clicking the left button. The right button works as ESC.

The data entry is done in specific windows and the movement of the cursor within the window is done with the mouse or the keyboard. The data entered in the window is accepted with ENTER or with a mouse click on the OK located in the bottom of the window.

---

## Creating Drawings

The first step to develop an application is to create the drawings that will be used as the background in the application's screens. These drawings can be the representation of a specific process or machine or can be the frames for tables or messages.

You can create the drawings with any Graphics Generator compatible with Windows BMP (16 color). A good option is to use Paintbrush which is provided together with Windows.

After creating a drawing and saving it as a 16 color Bitmap, you have to convert it to the VGC file which is the format used by Power Assist while running the application. The VGC format is a compressed bitmap file. The conversion is made by the program DIBVGC.EXE. To run it just type:

```
dibvgc <filename.BMP> ENTER
```



The drawings must be created using 640 x 464 pixel size and must be saved as 16 color Bitmaps. If you are using Paintbrush you can define the drawing size selecting *Options/Image Attributes*.

**If your Windows is configured for more than 16 colors:** when saving the drawing, change the option *Save File as Type* to 16 color Bitmap.



**Tip:** When creating frames for Power Assist's objects Table, Text and Setpoints you must consider that Power Assist inserts them respecting rows and columns (80 x 30). Every character occupies 16 x 8 pixel. The easiest way to create frames to this objects is: create the screen with Table, Text and Setpoints using Power Assist; import the screen to Paintbrush; and create the frames.



**Tip:** If you have a Graphics Generator that is not compatible with Windows you may try to do the following:

- a) Run your Graphics Generator under Windows.
- b) Load the drawing.
- c) Use the key PRINT SCREEN to send the drawing to the Windows Clipboard.
- d) Run Paintbrush and select Zoom Out.
- e) Get the drawing from the Clipboard.
- f) Save it using Windows .BMP format.

## Paintbrush Tips

A good option for creating drawings is to use Windows Paintbrush. The instructions on how to use Paintbrush are included in the Microsoft Windows User Guide. The following are some tips that will help you get better results from Paintbrush.

If you press the Shift key while moving the mouse:

The tools Line and Polygon allow only the drawing of horizontal, vertical and 45 degrees lines.

The tools Airbrush, Color Eraser, Eraser and Brush allow only horizontal and vertical movement of the cursor.

The tools Box, Filled Box, Rounded Box and Filled Rounded Box allow the drawing of perfect squares.

The tools Circle/Ellipse and Filled Circle/Ellipse allow the drawing of perfect circles.

The tool Pick defines a perfect square area.

After defining an area with the tools Scissors or Pick, you can copy this area using the key Ctrl and the mouse for drag and drop.

To define the foreground color, position the cursor over the desired color in the palette and click the mouse left button. To define the background color, do the same with the mouse right button.

A double click on:

The tool Pick is equivalent to the menu option View Picture.

The tool Eraser is equivalent to the menu option File New.

The tool Brush enables the editing of the brush shape.

The tool Color Erase changes all the drawing colors that are equivalent to the current foreground color to the current background color.

A palette color enables the editing of this color.

## Using Power Assist Development

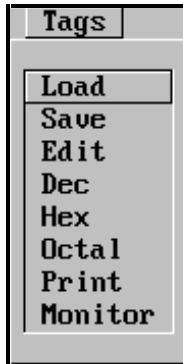
The main menu of Power Assist provides five options:

**Tags**  
**Insert**  
**Edit**  
**Screens**  
**Settings**  
**Application**

Power Assist uses a special window in the left of the screen every time you have to select a file. To select the file, point and click with the mouse or type the name of the file.

## Tags

The module *Tags* allows the user to configure the variables used in a specific application. Every variable is called a tag. These tags may be field tags or internal tags. Field tags are pressures, temperatures, etc. Internal tags are averages, operations, day, month, etc. The tags used in a specific application are defined in a tags file. The options of the Tags module are:



### Load

Loads tags files. Power Assist uses a special window on the left of the screen every time you need to select a file. To select a file, point and click with the mouse or type the name of the file.

### Save

Saves the tags file that is being edited.

## Edit

Edits Tags files. The first window asks the following options:

**Number of drivers:** Defines the number of I/O drivers used in the application to communicate with the data acquisition equipment (max. 6, min. 1).

**Number of blocks:** Defines the number of blocks of data used to communicate with the data acquisition equipment. These blocks are used to send a group of variables together to the PC, enhancing the communication performance.

**Number of text tags:** Defines the number of tags used to store strings.

**Number of mirror files:** Defines the number of mirror files used by the application. These files are used to get the values of tags available in another Power Assist running in another PC. It works according to the following description: Power Assist is able to generate a mirror file containing the current value of some of its tags. When it is running under a DOS compatible Network, Power Assist is able to access the mirror file generated by Power Assist running in another PC.

**Number of tags:** Defines the number of numerical tags used by the application.

**Number of expressions:** Defines the number of arithmetical used by the tags.

**Number of groups:** Defines the number of groups of tags. Each group may have a different scan mode: a) request: the value of the tag is read from the data acquisition equipment every time it is requested by a trend, alarm, etc.; b) time: the value of the tag is read from the data acquisition equipment according to a defined cycle; c) event: the value of the tag is read from the data acquisition equipment when an event is detected (see the function *Read Group*).

**Number of retries:** Defines the number of retries that should be made when the communication has failed.

Power Assist shows a window at the upper left corner of the screen indicating the I/O drivers loaded in the PC memory. Each driver has its own identification number.

### ***Drivers Configuration Window***

The next window is used for I/O driver configuration.

**Drv:** Defines the number of each I/O driver that will be used.

**Parameters p1 to p4:** These are parameters used to configure each I/O driver. You have to check the DOC file of each driver that will be used to see the usage of the parameters. As an example, imagine we are using the data

acquisition equipment XYZ whose I/O driver is XYZ.EXE. Its DOC file, XYZ.DOC, shows that:

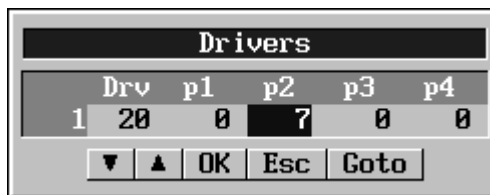
```
p1 = serial port
    0 - COM1
    1 - COM2
    2 - COM3
    3 - COM4

p2 = baud rate

    0 - 110
    1 - 300
    2 - 600
    3 - 1200
    4 - 2400
    5 - 4800
    6 - 9600
    7 - 19600

p3 = not used
p4 = not used
```

If you load the XYZ driver before running Power Assist, when you select the option Edit-Tags you will see a window at the upper left corner of the screen indicating that the XYZ driver is loaded and that its identification number (as an example) is 20. Imagine that you will use the serial port COM1 and that the baud rate will be 19600. The Drivers Configuration Window will be as follow:



## **Blocks Window**

If the number of blocks is different than zero the next window is used to configure the blocks.

Every line defines one block:

**Drv:** Defines the I/O driver number for the block. If you are using more than one I/O driver, you can select the correct driver for each block clicking the mouse over the Drv field or using the Space Bar.

**Parameters b1 to b4:** Used to define the address for each block. To see the usage of these parameters for a specific driver check in the corresponding DOC file. Using the example above, imagine that XYZ.DOC shows that:

b1 - PLC number in the network

b2 - address of the first variable of the block

b3 - number of variables to be read

b4 - not used

In our example you will define two blocks of data coming from the PLC whose number is 1. The first block will start at the address 1000 and will have 50 elements (variables). The second will start at the address 1500 and will have 100 elements.

Block					
	Drv	b1	b2	b3	b4
1	20	1	1000	50	0
2	20	1	1500	100	0

▼ ▲ OK Esc Goto

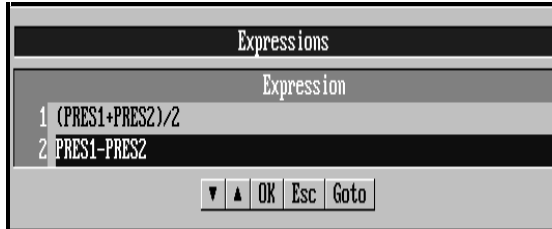
The number (index) of the block is indicated on the left part of the window. In the example above we have two blocks whose indexes (numbers) are 1 and 2.

### ***Text Tags Window***

The next window defines the names of the text tags used by the application. These text tags are used to store the name of the operator, directory, recipe, etc. The first two tags are reserved (DIRET and DIREC). The first one (DIRET) is used to define the directory where the batch histories are saved. The second one (DIREC) is used to define the directory to save the recipes.

## Expressions Window

This window is used to define arithmetical expressions. Each expression is defined in one line. The following operations are available: “+”, “-”, “\*”, “/”, “SQR”, “^”, “LOG” and “%” (remainder of the division). One example with two expressions is:



The first expression calculates the average of the tags PRES1 and PRES2. The second expression subtracts PRES2 from PRES1. The result of the expression is assigned to the tag using the operation *Exp*, described in the section *Tags Window*. The number (index) of the expression is indicated on the left part of the window.

## Groups of Tags Window

Tags can be separated in different groups according to the scan mode and/or scan cycle. The scan modes are:

**Request:** the value of the tag is read from the data acquisition equipment every time it is requested by a trend, alarm, etc.;

**Time:** the value of the tag is read from the data acquisition equipment according to a defined cycle

**Event:** the value of the tag is read from the data acquisition equipment when an event is detected (see the function *Read Group* in the section *Application-Edit*).

In the example below, only one group was defined. The group scans the tag value using the mode *Request*. The option *Cycle* is only used for the mode *Time*. The number (index) of the group is indicated on the left part of the window.



The mode time works only when running the application (but does not work when monitoring the screen).

### ***Mirror Files Window***

If the number of mirror files is different than zero the next window is used to define each mirror file. You have to enter the path and the name of each file Power Assist is going to access. The number (index) of the file is indicated on the left part of the window. Values in the mirror file are assigned to tags using the operation *Mirror*, described in the section *Tags Window*.

As an example, imagine we have two mirror files:

Mirrors		
	File	Cycle ASCII
1	F:\APP1\MIRROR1	1 No
2	G:\APP2\MIRROR2	2 Yes

▼ ▲ OK Esc Goto

The option *Cycle* defines the cycle (in seconds) for reading the mirror file. The option *ASCII* indicates if the mirror file is an ASCII file.

## Tags Window

The next window is used to configure the numerical tags. This window shows 11 tags at one time. You can see the other tags using the keys PgUp and PgDn or clicking the buttons with the arrows up and down. To move to one specific tag you can use the button GOTO, specifying the tag index (number). This option is very useful when you have a large number of tags in your application. The number (index) of the tag is shown on the left of the tag name.

Tags <TMP>											
	Name	Oper	Drv	n1	n2	n3	n4	Gp	?	Alfa	Beta
1	TAG0001	PLC	0	0	0	0	0	1	No	0.0000	0.0000
2	TAG0002	PLC	0	0	0	0	0	1	No	0.0000	0.0000
3	TAG0003	PLC	0	0	0	0	0	1	No	0.0000	0.0000
4	TAG0004	PLC	0	0	0	0	0	1	No	0.0000	0.0000
5	TAG0005	PLC	0	0	0	0	0	1	No	0.0000	0.0000
6	TAG0006	PLC	0	0	0	0	0	1	No	0.0000	0.0000
7	TAG0007	PLC	0	0	0	0	0	1	No	0.0000	0.0000
8	TAG0008	PLC	0	0	0	0	0	1	No	0.0000	0.0000
9	TAG0009	PLC	0	0	0	0	0	1	No	0.0000	0.0000

▼ ▲ OK Esc Goto

**Name:** Defines the name of the tag.

**Oper:** Indicates from where the value of the tag comes. The options are:

**PLC:** this option indicates that the value of the tag comes from data acquisition equipment. The I/O driver is indicated in the column DRV. If you are using more than one I/O driver, you can select the

correct driver for each tag by clicking the mouse over the Drv field or using the Space Bar. The parameters n1 to n4 indicate the address of the tag in the equipment and are explained in the specific DOC file of the driver. Using the DOC file below to access the variable whose address is 1230 in the PLC 1 you have to define n1 as 1 and n2 as 1230:

n1 - PLC number in the network
n2 - address of the variable to be read
n3 - not used
n4 - not used

**Block:** gets the value of the tag from the block of data received from the data acquisition equipment. The column n1 indicates the number of the block and n2 the element of the block. Using the example described in the configuration of blocks, to access the variable whose address is 1020 you have to define n1 as 1 (block 1 - starting at the address 1000) and n2 as 21 (element 21 - address 1020). To access the variable whose address is 1500 you have to define n1 as 2 and n2 as 1.

**Sum:** sums two or more tags. n1 indicates the number (index) of the first tag and n2 the number of the last tag.

**-:** subtracts the tag indicated in n1 from the tag indicated in n1.

**\*:** multiplies the tag indicated in n1 by the tag indicated in n2.

**/:** divides the tag indicated in n1 by the tag indicated in n2.

**Rem:** gets the remainder of the division of the tag indicated in n1 by the tag indicated in n2.

**^2:** calculates the power of 2 of the tag indicated in n1.

**Sqr:** calculates the square root of the tag indicated in n1.

**Avg:** calculates the average of two or more tags. n1 indicates the first and n2 the last tag.

**Flash:** changes the value of the tag from 0 to 20000 every half second if the tag with the number indicated in n1 is different than zero.

**Day:** day of the month.

**Month:** month.

**Year:** year.

**Hour:** hour of the day .

**Min:** minute.

**Sec:** second.

**Const:** assigns a constant value to the tag. The constant value must be defined in n1.

**Temp:** this option allows the user to set the value of an internal tag (that is not sent to the data acquisition equipment) with the option *Setpoint* of the module *Insert*. It may be used to give a number to a product or a batch.

**And:** makes a logical AND with the tags indicated in n1 and n2.

**Or:** makes a logical OR with the tags indicated in n1 and n2.

**Xor:** makes a logical XOR with the tags indicated in n1 and n2.

**Not:** makes a logical NOT with the tag indicated in n1.

**Bit:** gets one bit from a tag. n1 indicates the tag number (index) and n2 the bit number. As an example, imagine you have the tag ABC whose number (index) is 10. If you want to extract the first bit of this tag you have to define n1 as 10 and n2 as 0.

**Bbit:** gets one bit from a tag directly from a block. n1 indicates the block number, n2 the element in the block and n3 the bit number. Using the example described in the configuration of blocks, to get the second bit of the variable whose address is 1020 you have to define n1 as 1, n2 as 21 and n3 as 1.

**Ini\_v:** registers the value of a tag at the beginning of the batch. n1 indicates the tag number. As an example, imagine you have the tag ABC whose number (index) is 10. If you want to register the value of the tag ABC at the beginning of a batch you have to create another tag, define its Oper as *Ini\_v* and n1 as 10. See also the function *Start Batch* in the section *Application-Edit*.

**Fin\_v:** registers the value of a tag at the end of the batch. n1 indicates the tag number. As an example, imagine you have the tag ABC whose number (index) is 10. If you want to register the value of the tag ABC at the end of a batch you have to create another tag, define its Oper as *Fin\_v* and n1 as 10. See also the function *End Batch* in the section *Application-Edit*.

**Alarm:** returns 1 if there is any alarm that has not been acknowledged.

**Proce:** used for batch process, returns 1 if the batch has been started. n1 indicates the number of the history. See also the function *Start Batch* in the section *Application-Edit*.

**DayW:** gives the day of the week.

**+:** adds the tag indicated in n1 to the tag indicated in n2.

**Ln:** gets the natural logarithm of the tag indicated in n1.

**Log:** gets the logarithm (base 10) of the tag indicated in n1.

**Count:** returns the elapsed seconds after the application was started. n1 must contain the number of the counter. May be reset by the function *Set tag* or by a *Setpoint*.

**>:** returns 1 if the tag indicated in n1 is greater than the tag indicated in n2.

<: returns 1 if the tag indicated in n1 is lesser than the tag indicated in n2.

=: returns 1 if the tag indicated in n1 is equal to the tag indicated in n2.

^: returns the amount of the tag indicated in n1 to the power of the tag indicated in n2.

**Mirror:** used to indicate tags contained in mirror files. n1 indicates the number (index) of the mirror file and n2 the number (index) of the tag in the mirror file. Using the example described in the configuration of mirror files, if you want to access the second tag of the mirror file *F:\APP1\MIRROR1* you have to define n1 as 1 and n2 as 2.

**Index:** used to index tags. n1 indicates a tag which value is used to point to another tag. Let's have an example where we have the tags TAG-01, TAG-02, TAG-03 and TAG-04. The number (index) of this tags are 1, 2, 3 and 4. Let's assume that n1 in TAG-01 is defined as 2, pointing to TAG-02, whose number (index) is 2. When the value of TAG-02 is 3, it will point to TAG-03, whose number (index) is 3. In this case the value of TAG-03 will be assigned to TAG-01. If the value of TAG-02 is 4, the value of TAG-04 will be assigned to TAG-01.

**Al-low:** used to change the lower limit for an alarm. n1 indicates the number (index) of the alarm.

**Al-up:** used to change the upper limit value for an alarm. n1 indicates the number (index) of the alarm.

**I-Plc:** another way of indexing tags. Its usage is similar to the one of the operation *PLC*. In the operation *I-Plc*, n1, n2, n3 and n4 are not fixed values, they point to tags whose values are used to perform the operation *PLC*.

**Exp:** returns the result of an expression. n1 indicates the number (index) of the expression.

To select an option you can use the mouse, the Space Bar, Ctrl →, Ctrl ←, or type it.

**Drv:** when using the option *PLC* this column defines the I/O driver used. If you are using more than one I/O driver, you can select the correct driver for each tag clicking the mouse over the *Drv* field or using the Space Bar.

**Parameters n1 to n4:** when using the option *PLC* this column defines the address of the tag in the data acquisition equipment. The parameters n1 to n4 indicate the address of the tag in the equipment and are explained in the specific ".DOC" file of the driver.

Other operations use the parameters n1 to n4 to point to other tags. For instance, in the operation *Sqr*, n1 indicates the number (index) of the tag which value will be used in the square root operation.

**Gp:** indicates the group number for the tag.

**?:** this option activates the using of ALFA & BETA.

**Alfa & Beta:** these columns are used to make the scaling of the tag value. The number in ALFA multiplies the tag value. The number in BETA is added to the tag value.

**Dec**

Changes the base of the numbers used in the edition of the Tags to decimal.

**Hex**

Changes the base of the numbers used in the edition of the Tags to hexadecimal. This option is used when you want to enter the parameters n1 to n4 using hexadecimal numbers.

**Octal**

Changes the base of the numbers used in the edition of the Tags to octal. This option is used when you want to enter the parameters n1 to n4 using octal numbers.

**Print**

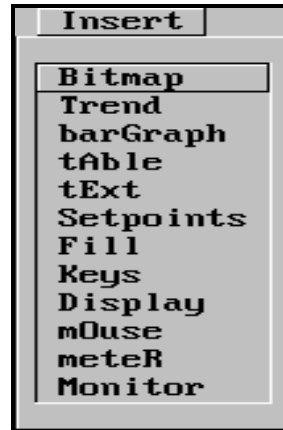
Prints the tags configuration.

**Monitor**

Monitors the current screen.

## Insert

This module is used to configure screens. It contains the following objects:



The Bitmap is used as a background for the screen and can range from plain gray to a complex drawing of the factory (see the section Creating Drawings).

The other objects, with the exception of Fill and Keys, are inserted in rectangular areas on the screen. Their position is defined by the two extreme corners of the rectangle, which are marked by a click of the left mouse button or with the arrow keys and ENTER. Fill is also positioned with a click of the left mouse button or with the arrow keys and ENTER. The default movement step of the cursor is pixel by pixel. It can be changed to character by character (8 x 16 pixel) with the F9 key, making it easier to align objects.

The objects Table, Text and Setpoints use the regular text font (8 x 16 pixel) and are positioned on the screen with the character by character movement step of the cursor.



Other objects use various text fonts so do not forget to include the available fonts in your working directory.

### **Bitmap**

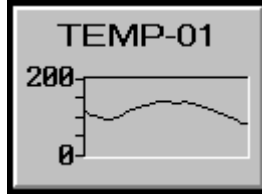
The first step to create an application is to make the drawings that will be used by Power Assist. You can create the drawings in any Graphics Editor compatible with Windows .BMP. While running, Power Assist uses a compressed Bitmap file whose extension is ".VGC". The conversion is made by the program DIBVGC.EXE. To run it just type:

```
dibvgc <filename.BMP> ENTER
```

The option *Bitmap* in the module *Insert* is used to define which drawing (bitmap file) will be used on the screen that is being configured.

---

## Trend



This object is used to configure trends. After positioning the object on the screen you have to enter the following options:

**Number of values:** number of tag values (samples) in the trend.

**Lower limit:** lower limit of the trend.

**Upper limit:** upper limit of the trend.

**Background:** background color of the trend.

**Number of tags:** number of tags in the trend (max. 5).

**Frame (Y/N):** defines if the object has a frame.

**Text (Top/Bottom/Off):** defines the position of the text in the frame.

**Scale:** defines if the frame includes a scale.

**Font:** defines the font used in the frame text.

**Frame color:** defines the frame color.

**Text color:** defines the text color.

**Precision:** defines the number of decimal digits for the scale text.

**Scale divisions:** number of divisions in the scale.

**Text:** text used in the frame.

**Cycle:** defines the scan cycle (in seconds). As an example, if you define number of values equal to 120 and cycle equal to 1 second, the trend will show the values of the last 2 minutes.

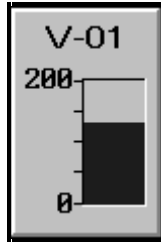
**History (Y/N):** defines if the values of the tags in the trend will come from a history file. If this option is selected, the trend is always initialized with the values contained in a history file.

**History number:** if the option history is selected, defines the number of the history used. As an example, imagine that you define a trend with 60 values, selecting the option history with the values from a history file that is being updated every 2 seconds. Every time you enter the screen, the trend is initialized with the last 60 values saved to the history file (120 seconds). When a new value is saved to the history file, the trend is also updated.

To change the colors, position the cursor over the color (with keyboard or mouse) and select the desired color in the color pad (with keyboard or mouse). The window is confirmed with the mouse or *ENTER*.

The next window defines the tags used in the trend. The name of the tag can be typed or can be selected, from the list entered in the tags edition, using the mouse, Space Bar or **Ctrl →** and **Ctrl ←**.

## Bargraph



Defines a bargraph. After positioning the object on the screen you have to enter the following options:

**Number of bars:** number of bars (tags) represented in the bar graph (max.10).

**Lower limit:** lower limit of the bargraph.

**Upper limit:** upper limit of the bargraph

**Position:** defines vertical or horizontal bargraph.

**Background:** background color of the bargraph.

**Frame (Y/N):** defines if the object includes a frame.

**Text (Top/Bottom/Off):** defines the position of the text in the frame.

**Scale:** defines if the frame has scale.

**Font:** defines the font used in the frame text.

**Frame color:** defines the frame color.

**Text color:** defines the text color.

**Precision:** defines the number of decimal digits for the scale.

**Scale divisions:** number of divisions in the scale text.

**Text:** text used in the frame.

To change the colors, position the cursor over the color (with keyboard or mouse) and select the desired color in the color pad (with keyboard or mouse). The window is confirmed with the mouse or *ENTER*. The next window defines the tags used in the bargraph. The name of the tag can be typed or can be selected from the list entered in the tags edition, using the mouse, Space Bar or **Ctrl →** and **Ctrl ←**.



**Tip:** One interesting technique is to combine two or more bargraphs (without frame) for the same tag, using different colors and creating the impression that you have only one bargraph with multiple colors. For instance you can create a bargraph for TEMP-01 defining the lower limit as 0, the upper limit as 100 and the color as green. You then create a second bargraph for the same tag on the top of the first one defining the lower limit as 100, the upper limit as 200 and the color as red.

---

## Table

Used to show the value of one or more tags. This object uses the regular text font (8x16 pixel) and is positioned on the screen with the character by character step of the cursor. The number of tags contained in the table will be proportional to the number of lines contained in the area defined (1 line = 16 pixel). After positioning the object on the screen you have to enter the following options:

**On/Off:** used to show digital values.

**Background:** background color of the value(s).

**Precision:** number of decimal digits (max. 4).

**Fill:** defines if the rest of the area will be filled with *Blanks* or *Zeros*.

## Text

Used to define texts. This object uses the regular text font (8 x 16 pixel) and is positioned on the screen with the character by character step of the cursor. The number of lines of texts contained in the object will be proportional to the number of lines contained in the area defined (1 line = 16 pixel). After positioning the object on the screen you have to enter the following options:

**Alignment:** defines the aligned of the text: left, right or centralized.

**Foreground:** defines the color of the text.

**Background:** defines the background color.

**Type:**

**Fixed:** used for a text that does not change (ex. “Pressure-01”).

**Tag:** used to change the value of text tags (ex. name of the recipe). Every line is related to one text tag.

**Messages:** defines messages that appear according to tag values (ex. “high” if the tag PRES-01 > 100 and “low” if the tag PRES-01 < 100).

**Display:** used to show the value of text tags. Very similar to the option *Tag*, but does not allow data-entry.

**Verification Tag:** defines the tag used to set the messages. Valid only for the option *Message*.

**Number of zones:** defines the number of zones to set messages (max. 6). Each zone will be related to one message. Valid only for the option *Message*.

**Group:** when you save a recipe, the setpoints and text tags contained in the current screen are saved to disk. You can divide the setpoints and text tags in groups and relate the recipe to only part of the setpoints and text tags contained in the screen. Valid only for the option *Tag*.

If the option used is *Tag* or *Display*, the next window is used to select the tag(s). Remember that every line contained in the area defined for the object is related to one text tag.

If the option is *Message*, the next window is used to define the message zones:

**Low:** Lower limit of the zone.

**Up:** Upper limit of the zone.

**Foreg:** Defines the text color in the zone.

**Backg:** Defines the background color in the zone.

**Text:** Defines the text in the zone.

If you are using the option *Fixed*, after confirming the window you have to type the desired text.

### Setpoints

This option enables the user to send setpoints to the data acquisition equipment. This object uses the regular text font (8 x 16 pixel) and is positioned on the screen with the character by character step of the cursor. The number of tags contained in the object will be proportional to the number of lines contained in the area defined (1 line = 16 pixel). After positioning the object on the screen you have to enter the following options:

**Lower limit:** lower limit accepted by the setpoints.

**Upper limit:** upper limit accepted by the setpoints.

**Precision:** number of decimal digits (max. 4).

**Foreground color:** defines the setpoints' color.

**Background color:** defines the background color.

**Foregr entry color:** defines the setpoints' color during the data entry.

**Backgr entry color:** defines the background color during the data entry.

**Initial Values:** defines how the values of the setpoints are going to be initialized when the screen is loaded:

**Plc:** initializes the setpoints with the value of the tags in the data acquisition equipment. This option indicates that when the screen is loaded the values of the tags related to the setpoints are read from the data acquisition equipment and shown in the setpoints objects on the screen.

**Disk:** this option indicates that when the screen is loaded the setpoints are initialized with the values previously assigned to these setpoints in this screen.

**Zero:** initializes the value of the setpoints with zero.

**Automatic refresh:** defines if the setpoint will be sent to the data acquisition equipment immediately after being typed or will wait for the function *Send Setpoints*.

**On/Off:** is used to change the value of the tag to 1 or zero.

**Group:** when you save a recipe, the setpoints and text tags contained in the current screen are saved to disk. You can divide the setpoints and text tags in groups and relate the recipe to only parts of the setpoints and text tags contained in the screen. You can also use the groups with the function *Send Setpoints* to define which setpoints are going to be sent to the data acquisition equipment.

**Data entry:** defines if the setpoint allows data entry.

## Fill

This option is used to change the color of one area according to the value of one tag. First you need to mark a point within the area you want to change the color. After positioning the object on the screen you have to enter the following options:

**Verification tag:** defines the tag whose value controls the fill.

**Border color:** defines the color that limit the area whose color is changed.

**Background color:** defines the color for the area when the tag value is out of the defined zones.

**Number of zones:** defines the number of zones.

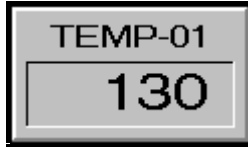
**Flash cycle:** defines the flash cycle.

The next window is used to define the lower and upper limits for the zone, the associated color and the option that makes the color flash (blink).

## Keys

Defines the local function of the keys. The configuration of the keys will be explained in the section *Application-Edit*.

## Display



This option is similar to the option Tables. The difference is that it uses multiple fonts to show the tag value. After positioning the object on the screen you have to enter the following options:

**Tag:** defines the tag in the display.

**Digits:** defines the number of digits used.

**Precision:** number of decimal digits (max. 4).

**Foreground:** defines the foreground color.

**Background:** defines the background color.

**Font:** defines the text font used.

**Frame (Y/N):** defines the display with or without frame.

**Text (Top/Bottom/Off):** defines the position of the text in the frame.

**Font:** defines the font used in the frame text.

**Frame color:** defines the frame color.

**Text color:** defines the text color.

**Text:** text used in the frame.

## Mouse



Defines an area on the screen that is going to be mouse sensitive. This area is associated with an action like changing the screen, setting the value of a tag, etc. After positioning the object on the screen you have to enter the following options:

**Type:** defines the type of the action. The valid types are:

**Function:** When pointing and clicking with the mouse on the area it will be like pressing the key defined in the option *Key*.

**Tag-Momentary:** The value defined in option *Value on* is assigned to the tag defined in the option *Tag*.

**Tag-Toggle:** This type works like on/off buttons. When pointing and clicking with the mouse on the area, the value defined in the option *Value on* is assigned to the tag defined in the option *Tag*. In the next click, the value defined in the option *Value off* is assigned to the tag defined in the option *Tag*.

**Tag-Jog:** The value defined in the option *Value on* is assigned to the tag defined in the option *Tag* when the mouse left button is pressed. When the button is released, the value defined in the option *Value off* is assigned to the tag defined in the option *Tag*.

**Key:** defines the function key associated to the mouse sensitive area. Valid only for the type *Function*.

**Button:** defines if the area is represented as a button. If not a button it can be used to create a mouse sensitive area over a drawing (ex. factory).

**Text:** defines if the area includes a text. Usually used with buttons.

**Foreground:** defines text color.

**Border:** defines the number of pixels in the button border.

**Font:** defines the text font used.

**Text off:** defines the text that is showed when the button is not pressed.

**Text on:** defines the text that is showed when the button is pressed.

**Background off:** defines the background color that is used when the button is not pressed.

**Background on:** defines the background color that is used when the button is pressed.

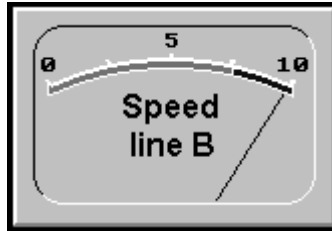
**Value off:** used to set the value of the tag.

**Value on:** used to set the value of the tag.



If you are using the option *Text*, do not forget to include the text fonts in the directory where you run Power Assist.

## Meter



This object is an analog display (meter). After positioning the object on the screen you have to enter the following options:

**Tag:** defines the tag used in the meter.

**Double:** defines the needle width (single or double).

**Background:** defines the meter with or without background.

**Foreground color:** defines the color of the needle.

**Background color:** defines the background color of the meter.

**Lower limit:** defines the lower limit of the meter.

**Upper limit:** defines the upper limit of the meter.

**Frame (Y/N):** defines the meter with or without frame.

**Text (Top/Bottom/Off):** defines the position of the text in the frame.

**Font:** defines the font used in the frame text.

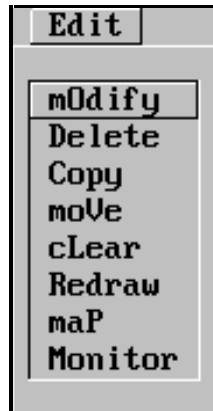
**Frame color:** defines the frame color.

**Text color:** defines the text color.

**Text:** text used in the frame.

## Edit

This module is used to edit the current screen:



### Modify

Modifies objects on the screen (trends, bargraphs, etc.)

### Delete

Deletes objects.

### Copy

Copies an object.

### Move

Moves an object.

### Clear

Clears the current screen.

---

## Monitor

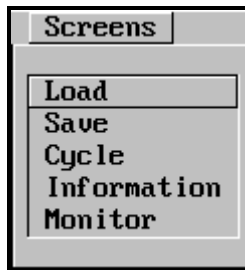
Monitors the current screen.



**Tip:** The best way to re-size objects is to go to *Edit-Modify*, selecting the object to re-size (all the settings of that object will be the default settings when you create another object). You can then delete the object and create another one with the desired size without having to re-enter all the settings.

## Screens

Used to manipulate screen files. Has these options:



### Load

Loads a screen file.

### Save

Saves the current screen.

## Cycle

Sets the cycle for screen update in hundreds of seconds. As an example if you set it to 200, all the objects in the screen (trends, tables, etc.) will be updated every 2 seconds.

## Information

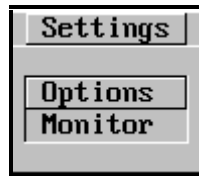
Gives general information about the current application.

## Monitor

Monitors the current screen.

## Settings

This module is used for the general settings of the software:



## Options

**Black fonts (Y/N):** defines if the menu fonts are black. The default color is white.

**Keypad (Y/N):** defines if the keypad for data entry with mouse or touchscreen is active.

**MM/DD/YY (Y/N):** used to set the date format.

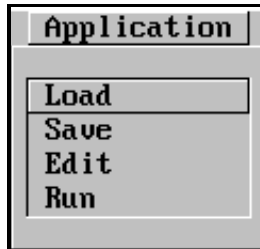
---

## Monitor

Monitors the current screen.

## Application

This module is used to make the general configuration of the application:



### Load

Loads application files.

### Save

Saves the current application file.

### Edit

This option is used to configure applications. The first window has the following options:

**Tags file:** defines the tags file used by the application.

**Mirror file:** defines the name of the file where the mirror values will be stored.

**First screen:** defines the first screen used by the application.

**Mirror cycle:** defines the cycle for writing to the mirror file. If you set it to 5, the mirror file will be updated every five seconds.

**Type of mirror:** defines the write mode for the mirror file:

**Time:** the mirror file is updated according to a defined cycle.

**Event:** the mirror file is updated when an event is detected (see the function *Write Mirror*).

**Mirror file ASCII:** Defines if the mirror file is ASCII or binary. Binary files have better performance. ASCII files allow data exchange with other software.

**Alarms and events:** defines the number of tags for alarms and events. The alarms and events are defined in the *Alarms and Events Window*.

**Mode (On/Off):** defines if the verification of alarms and events will be active.

**Tags for mirror:** defines the number of tags saved to the mirror file. The tags saved to the mirror file are defined in the *Mirror Tags Window*.

**Mode (On/Off):** defines if the application will generate a mirror file.

**Alarm high color:** defines the background color used in the window for alarms acknowledgment. It is also used as a background color for the object *Table* (for the tags in alarm).

**Alarm low color:** defines the background color used in the window for alarms acknowledgment. It is also used as a background color for the object *Table* (for the tags in alarm).

**Communic. alarm color:** defines the background color used in the window for alarms acknowledgment. It is also used as a background color for the object *Table* (for the tags in alarm). Used for bad communication alarm.

**Sound warning (Y/N):** sets the sound warning associated with the detection of alarms.

**Visual warning (S/N):** sets the visual warning associated with the detection of alarms. The visual warning is a small window that appears in the center of the screen indicating the presence of alarms.

**Warning cycle:** defines the cycle for visual and sound warning.

**Alarms file size:** defines the number of registers in the alarms file. As an example, if you define it as 500 you will have the last 500 events available in the alarms file.

**Events file size:** defines the number of registers in the events file. As an example, if you define it as 100 you will have the last 100 events available in the events file.

**Histories:** defines the number of histories (Power Assist supports multiple histories). You can group the tags in different histories according to sample interval, continuous or batch process, etc.

**Alarms to printer (Y/N):** defines is the alarms are sent to the printer as soon as they are detected.

**Printer for alarms:** defines the printer used for printing the alarms.

### **Functions Window**

The next window is used to define the functions associates with the global keys. The local keys, defined in each screen, have priority over the global keys. The first column of the window is used to define the keys, using the mouse or typing it. The valid keys are:

F1 to F12  
Shift F1 to Shift F12  
Ctrl F1 to Ctrl F12  
Alt F1 to Alt F12  
Esc  
A to Z

In the second column you define the function associated to the key. In the third column you may define a parameter for the function. One example would be the name of the screen for the function *Change Screen*.

The functions available are:

**Menu:** calls a pre-defined menu containing the most used functions.

**Historical Analysis:** makes an historical analysis. If the application has more than one history file you can use the column *TAG1* to define a tag whose value will indicate the history used by the function.

**Print History:** prints the historical analysis. If the application has more than one history file you can use the column *TAG1* to define a tag whose value will indicate the history used by the function.

**SPC:** Statistical Process Control. If the application has more than one history file you can use the column *TAG1* to define a tag whose value will indicate the history used by the function.

**Print Screen:** prints the current screen.

**Print Form:** prints a formatted report. The name of the format file should be defined in the column *Paramet*. See the section *Running the Application-Analyzing Data* to see how to create formatted reports.

**Acknowledge Alarms:** show the alarm log and makes the acknowledgment of alarms.

**Print Alarms:** prints the alarm log.

**Verify Events:** show the events log.

**Print Events:** prints the events log.

**Save Recipe:** saves the values of the setpoints and the text tags present in the current screen to a recipe file. The column *TAG1* may be used to define a tag whose value indicates the name of the recipe. The text tag DIREC may be used to define the directory where the recipe is going to be saved.

**Load Recipe:** loads a recipe file. The column *TAG1* may be used to define a tag whose value indicates the name of the recipe. The text tag DIREC may be used to define the directory from where the recipe is going to be loaded.

**Send Setpoints:** sends the setpoints present in the current screen to the data acquisition equipment.

**Confirm Setpoints:** sends setpoints with confirmation.

**Start Batch:** starts a batch. If the application has more than one history file you can use the column *TAG1* to define a tag whose value will indicate the history used by the function. The tag DIRET may be used to define the directory where the batch history file is going to be saved.

**End Batch:** ends a batch. If the application has more than one history file you can use the column *TAG1* to define a tag whose value will indicate the history used by the function.

**Set Tag:** sets the tag defined in the column *TAG1* with the value of the tag defined in the column *TAG2*. You can define a fixed value in the column *Paramet* to be used instead of *TAG2*

**Print Graph:** prints a graphical historical analysis. If the application has more than one history file you can use the column *TAG1* to define a tag whose value will indicate the history used by the function.

**Save Values:** saves the values of the tags to the historical values file. This function is used when the historical values mode is set to Events.

**Quit:** quits the application.

**Direct Alarms Ack.:** makes the acknowledgment of alarms.

**Print:** used to print or copy an file.

**Set Alarms:** used to change the alarm limits.

**Print SPC:** Prints Statistical Process Control. The column *TAG1* may be used to define a tag whose value indicates the history number.

**Change screen:** go to a different screen. The name of the screen should be defined in the column *Paramet*. If you define a tag in the column *TAG1*, the changing of the screen will only occur if the value of this tag is different than zero.

**Increment Tag:** increments the value of the tag defined in the column *TAG1* with the value of the tag defined in the column *TAG2*. You can define a fixed value in the column *Paramet* to be used instead of *TAG2*.

**Switch Tag:** switches the value of the tag defined in the column *TAG1* between zero and the value of the tag defined in the column *TAG2*. You can define a fixed value in the column *Paramet* to be used instead of *TAG2*.

**Refresh Setpoints:** used to refresh the values of the setpoints in the current screen (read from the data acquisition equipment).

**Read Group:** Reads a group of the tags from the data acquisition equipment. The number of the group is defined by the tag in the column *TAG1*.

**Write Mirror:** Updates the mirror file.

**Stop Counter:** Stop the counter whose number is defined in the column *Paramet*.

**Restart Counter:** Restart the counter whose number is defined in the column *Paramet*.

**Keypad:** Calls the keypad for data entry with mouse or touchscreen. You have to define in the column *TAG1* the tag to assign the value entered in the keypad. You can also send the value entered in the keypad to a setpoint, if you leave the column *TAG1* blank and call the keypad while the cursor is over the setpoint.



If you press spacebar in the columns *Key*, *Function*, *TAG1* or *TAG2*, the next option is selected (ex. F1 to F2). To clear the value of any of these columns, press DEL.

### **Histories Window**

The next window defines the histories:

**Name:** name of the history file. Used only for continuous process. For batch process there is one file for each batch and its name reflects the date and time of the moment when the batch is started.

**Tags:** number of tags in the history.

**Cycle:** cycle in seconds for saving the values to the history file.

**Process:** continuous or batch.

**Trigger:** defines if the values are saved to the history file according to the time cycle or to events (using the function *Save Values*).

**Size:** maximum number of values for each tag in the history.

The number (index) of the history is showed on the left part of the screen. Several functions as *Historical Analysis* and *Start Batch* use this number to identify which history will be used by the function.



It is not possible to modify the number of tags contained in an existing history file. If you attempt to do so, Power Assist will find an error when you run the application.

### ***History Tags and Mirror Tags Windows***

The next windows are used for the definition of the tags for each history file and for the mirror file.

### ***Alarms and Events Window***

The next window is the alarms and events definition:

**TAG:** defines the tags used to verify alarms or events.

**Al/Eve:** defines if it is an alarm or an event.

**Ana/Dig:** defines if it is a digital or an analog alarm.

**Al-Up e Al-Low:** used for analog alarms or events. Defines the alarm limits.

**Cycle:** defines the cycle (in seconds) for checking the alarm.

**Message:** defines the message associated with the alarm or event.

The number (index) of the alarm/event is shown on the left part of the window.

### ***Procedures Window***

The next window defines functions (actions) associated to alarms or events. The first column indicates the number (index) of the alarm or event. The second column indicates the function to be called.

### ***Printing Colors Window***

The last window is used to define the equivalent printing colors used in the function *Print Screen*.

### **Run**

Runs the application within PA.EXE. Power Assist Development enables the to run for only ten minutes.

## RUNNING THE APPLICATION

To run the application use the Runtime (PAX.EXE) with the following syntax:

PAX *application* [/v *n*] ENTER

*application* is the application file (.SHW).

*/v* - changes the interrupt used by Power Assist. Sometimes there is a conflict between the interrupts used by your PC and the one used by Power Assist. If Power Assist halts during its initialization there is a conflict. You can use this option to change the interrupt used by Power Assist.

*n* is the number of the interrupt (between 70 and 200).

## Setpoints and Text Tags

After loading a screen that has setpoints and text tags, the cursor will be positioned in the first setpoint or text entry (upper left corner). If the setpoints and the texts tags are aligned you can move the cursor with using the following keys:



You can also use the mouse or the Tab key to move the cursor.

## Alarms and Events

An alarm activates the following actions:

- a) If a tag was defined with the operation *Alarm*, its value is set to one.
- b) Changes the background color of the tag in alarm if this tag is shown as a table in the current screen.
- c) Logs the alarm to the alarms file.
- d) Starts the alarm warning.

## Alarms Acknowledgment

The alarms acknowledgment is made with the function *Acknowledge Alarms*. This function brings a window that shows the alarms log. The alarms that have not been acknowledged are marked with color configured in *Application-Edit*. The keys PgUp, PgDn, Ctrl-PgUp and Ctrl-PgDn are used for the window scroll. The key Enter makes the acknowledgment of the alarms.

## Events

Events are very similar to alarms. The only difference is that events do not have a warning.

## Analyzing Data

Power Assist is able to store the historical value of tags for further analysis. For this purpose it uses the following functions:

Function	Description	Output
Historical Analysis	makes the graphical historical analysis	video
Print History	makes numerical historical analysis (standard or formatted)	printer or file
SPC	statistical process control	video
Print SPC	statistical process control	printer
Print Form	generate formatted reports using the current tags value	printer or file
Print Graph	makes the graphical historical analysis	printer

---

## Historical Analysis

Power Assist allows the user to perform historical analysis of up to five tags at one time. This function has the following options:

- Tag:** defines the tag for SPC.
- Beginning date:** defines the beginning of the period.
- Starting time:** defines the beginning of the period.
- Ending date:** defines the end of the period.
- Ending time:** defines the end of the period.
- Y-Low:** defines the range for Y.
- Y-Up:** defines the range for Y.
- Select (Y/N):** defines if the file to be analyzed is other than the current one. This option is usually used with batch process because there is one history file for each batch.
- Directory:** defines the directory where the other file is.
- X/Y:** selects a X/Y graph.
- Auto-Range:** selects auto-range.
- Type:** defines the graph with lines or bars.

## Formatted Numerical Historical Analysis

The function *Print Historical Analysis* allows the user to print the historical values of the tags in a certain period of time. It has two modes: standard and formatted.

The standard mode prints the values according to a pre-defined format. It generates a printing whose header contains the names of the tags chosen and whose body shows the values of these tags (for the period requested).

Let's have an example where we want to see the values of the tags TEMP-01, TEMP-02 and PUMP-01 between 10:50:00 and 10:50:01. Let's assume that the cycle for saving to disk is 20 seconds (for the period requested we will have three values for each tag).

<i>Historical Analysis</i>				
From:	12/04/93 10:50:00			
To:	12/04/93 10:51:00			
<u>Date</u>	<u>Time</u>	<u>TEMP-01</u>	<u>TEMP-02</u>	<u>PUMP-01</u>
12/04/93	10:50:00	102.00	110.45	1
12/04/93	10:50:30	102.01	110.99	1
12/04/93	10:51:00	102.08	111.03	0

The formatted mode prints the values according to a format file (ASCII). This file is divided in two parts: header and body. The header defines the header format and the body defines the format of the output for each register (value).

If you include a tag in the header, the printing will show the current value of this tag.

The general controls to be included in the file are:

@I1@ Beginning of header  
@F1@ End of header  
@I2@ Beginning of body  
@F2@ End of body

Everything included between @I1@ and @F1@ will be considered header and everything included between @I2@ and @F2@ will be considered body. Anything outside these controls will be excluded from the printing.

The syntax for including tags in the header and in the body is:

**@%*[n.p]*f=Tag@**

- **@** Defines the beginning and the end of a field.
- **n** Defines the total number of digits.
- **p** Defines the number of decimal digits.
- **Tag** Defines the name of the *Tag*.

The syntax for including text tags in the header and in the body is:

**@%*[n]*s=Tag@**

- **@** Defines the beginning and the end of a field.
- **n** Defines the total number of digits.
- **Tag** Defines the name of the *Tag*.

The syntax for including messages related to tags value is:

**@Sn=Tag@**

**@val1@ @message1@**

**@val2@ @message2@**

...

**@valn@ @messagen@**

- **S** Indicates message.
- **n** Number of possible messages.
- **Tag** Defines the name of the *Tag*.
- **val1 to n** Values to be associated with messages.
- **message1 to n** Messages.

The syntax for including date and time in the body is:

**@%[n]s=REG\_DATE@ e**

**@%[n]s=REG\_TIME@**

- **n** Defines the total number of digits.
- **REG\_DATE** and **REG\_TIME**  
Reserved words.

Let's have an example using the same data of the last example:

```

@I1@

      EXAMPLE OF FORMATTED HISTORICAL ANALYSIS

Operator:      @%15s=OPER@
Date:  @%2s=MONTH@/@%2s=DAY@/@%2s=YEAR@

Date  Time    TEMP-01      TEMP-02    PUMP-01

@F1

@I2@@%8s=REG_DATE@ @%8sREG_TIME@ @%5.2f=TEMP-01@
@%5.2f=TEMP-02@    @S2=PUMP-01@ @1@ @ON@ @2@
@OFF@
@F2@
    
```

Let's assume that the text tag OPER currently has the string "Roger Smith" and the tags MONTH, DAY and YEAR have the values "12", "4" and "93". The format file above would generate the following printing:

<b>EXAMPLE OF FORMATTED HISTORICAL ANALYSIS</b>				
<i>Operator:</i>	<i>Roger Smith</i>			
<i>Date:</i>	<i>12/04/93</i>			
<i>Date</i>	<i>Time</i>	<i>TEMP-01</i>	<i>TEMP-02</i>	<i>PUMP-01</i>
<i>12/04/93</i>	<i>10:50:00</i>	<i>102.00</i>	<i>110.45</i>	<i>ON</i>
<i>12/04/93</i>	<i>10:50:30</i>	<i>102.01</i>	<i>110.99</i>	<i>ON</i>
<i>12/04/93</i>	<i>10:51:00</i>	<i>102.08</i>	<i>111.03</i>	<i>OFF</i>

---

## Statistical Process Control

The statistical process function has the following options:

<b>Tag:</b>	defines the tag for SPC.
<b>Chart type:</b>	R or S.
<b>Beginning date:</b>	defines the beginning of the period.
<b>Starting time:</b>	defines the beginning of the period.
<b>Ending date:</b>	defines the end of the period.
<b>Ending time:</b>	defines the end of the period.
<b>Y-up:</b>	defines the range for X-bar graph.
<b>Y-Low:</b>	defines the range for X-bar graph.
<b>LEL:</b>	defines the lower engineering limit required.
<b>UEL:</b>	defines the upper engineering limit required.
<b>Y-Up R/S:</b>	defines the range for R/S graph.
<b>Sub-group size:</b>	defines the size of the sub-group.
<b>Select (Y/N):</b>	defines if the file to be analyzed is other than the current one. This option is usually used with batch process because there is one history file for each batch.
<b>Directory:</b>	defines the directory where the other file is.

## Reports



Power Assist can generate process reports based in ASCII files containing the format of the reports. The tags are defined according to the following syntax:

**@tt[.nn]f|s=Tag@**

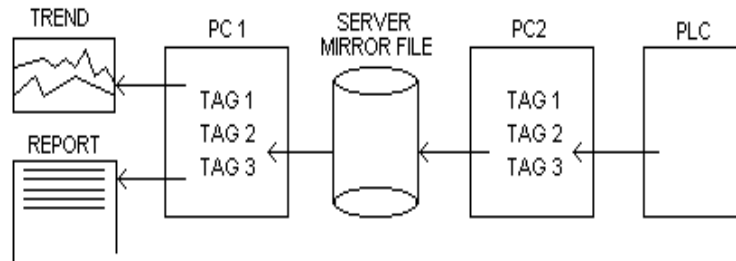
- **@** Defines the beginning and the end of a field.
- **tt** Defines the size of the field.
- **.nn** Defines the number of decimal digits.
- **f** Used for numerical tags.
- **s** Used for text tags (strings).
- **Tag** Defines the name of the tag.
- **@@** Form feed.

**Ex.:** Operator: **@40s=tag-001@**

In this example the value of the text tag will be added to the text "Operator:" in a field of 40 characters.

## Using a Network

Power Assist is able to exchange data between two or more applications running Power Assist in different PCs connected through a DOS compatible Local Network. To do so Power Assist uses files called Mirror Files.



## Set Tag

The value of a tag in the data acquisition equipment can be set using the object *Setpoints* or the functions *Set Tag*. When this function is called it sends the value of the tag defined in Tag2 to the tag defined in Tag1.

## Print

The function Print allows the user to copy and print ASCII files.

---

## Defaults

Some functions use default files that memorize the last options used in these functions. For instance when you use the function *Historical Analysis*, you have to enter some options like the range and the tags to be analyzed. These options are saved to a default file and are used the next time the function is accessed.

Power Assist uses the following default files:

DEFAULT.CHn <sup>1</sup>	Historical Analysis
DEFAULT.HPn	Print History
DEFAULT.CGn	Print Graph
DEFAULT.CEn	Statistical Process Control
DEFAULT.CFR	Print Form
DEFAULT.CAL	Print Alarms
DEFAULT.CEV	Print Events
DEFAULT.PR?	Print

These default files can also be used to call a function with the options pre-programmed. To do it just put the name of the file (without the extension) in the column *Paramet*. To have multiple defaults you just have to rename the file.

---

<sup>1</sup>n is the number of the history file.

## APPENDIX A - POWER ASSIST FILES



The following is a list of Power Assist files:

<b>PA.EXE:</b>	Development
<b>PAX.EXE:</b>	Runtime
<b>PADEMO.EXE:</b>	Demo
<b>POWER.INI:</b>	General settings of Power Assist (ex. menu colors)
<b>DIBVGC.EXE:</b>	Bitmap converter
<b>DEFAULT.CHn:</b>	Historical Analysis default
<b>DEFAULT.HPn:</b>	Print History default
<b>DEFAULT.CGn:</b>	Print Graph default
<b>DEFAULT.CEn:</b>	Statistical Process Control default
<b>DEFAULT.CFR:</b>	Print Form default
<b>DEFAULT.CAL:</b>	Print Alarms default
<b>DEFAULT.CEV:</b>	Print Events default
<b>DEFAULT.PRI:</b>	Print default
<b>*.LOG:</b>	Alarms and Events
<b>*.ACQ:</b>	Screens
<b>*.SHW:</b>	Applications
<b>*.TAG &amp; *.TGL:</b>	Tags
<b>*.TGS:</b>	Text tags
<b>*.VGC:</b>	Compressed (converted) bitmaps
<b>*.REC:</b>	Recipes
<b>*.*! &amp; *.-_:</b>	Histories.
<b>*.FON:</b>	Text fonts
<b>TITLE.PAC:</b>	Development name
<b>TITLE.PAR:</b>	Runtime name
<b>TITLE.PAD:</b>	Demo name

**This page is blank.**

## APPENDIX B - I/O DRIVERS

To communicate with data acquisition equipment Power Assist uses I/O drivers which are memory resident programs (TSR). These I/O drivers must be installed in the PC memory before running Power Assist.

Each I/O driver has the set of commands to communicate with a specific data acquisition equipment (or data acquisition equipment family).

### I/O Drivers available

The I/O drivers currently available are:

DRIVER	EQUIPMENT
3964	Siemens, Bosch or any other equipment using the 3964R protocol
ACROMAG	Acromag
CD600	Smar CD 600
GEFANUC	GE Fanuc using SNP protocol
HITACHI	Hitachi Hidic H-28DR
KM	Klockner Moeller
MBPLUS	Modicon or any other equipment using Modbus plus protocol
MODBUS	Modicon or any other equipment using Modbus protocol
PCLAB	Advantech PCLAB cards
PLC2	Allen Bradley PLC 2
PLC5-KG	Allen Bradley PLC 5 using KF2 or KE module
PLC5-KT	Allen Bradley PLC 5 using KT card
RELIANCE	Reliance using R-NET
SLC500	Allen Bradley SLC 500
SLC500KR	Allen Bradley SLC 500 using KR card
SLOT-PLC	Siemens Slot PLC
TELEMEC	Telemecanique using Unitelway

---

## Building an I/O driver

To develop an I/O driver for the Power Assist you just have to know the communication protocol of your data acquisition equipment and how to use the C language.

### Parameters

When you develop an I/O driver you have to define the usage of the following parameters:

*p1 to p4*: used to program the I/O driver.

Example:      p1 = 0, baud rate = 9600  
                 p1 = 1, baud rate = 19200

                 p2 = 0, using COM1  
                 p2 = 1, using COM2

                 p3 = 0, time out = 0.1 s  
                 p3 = 1, time out = 0.5 s

*b1 to b4*: used to define the address of the block of variables to be read in the data acquisition equipment.

Example:      b1, number of the PLC in the network  
                 b2, address of the first variable of the block  
                 b3, number of variables to be read  
*n1 to n4*: used to define the address of a variable to be read from and sent to the data acquisition equipment.

Example: n1, number of the PLC in the network n2, type of variable (0 = binary, 1 = BCD) n3, address of the variable

The usage of these parameters must be provided to the user of Power Assist.

## Functions

The driver must have the following functions:

*int get\_driver\_id()*

Power Assist uses a number to identify each I/O driver (ID). Power Assist calls this function every time it wants to know the number of an I/O driver that is resident in the memory. This function must return the I/O driver ID.

*char huge \*get\_driver\_name()*

This function gives the I/O driver name. Power Assist uses this function to show to its user the names of the I/O drivers that are resident in the memory. This function must return a string pointer. The variable that stores the I/O driver name must be declared as static.

*int start\_com(int p1, int p2, int p3, int p4)*

Power Assist uses this function to program the I/O driver (baud rate, time out, number of data acquisition equipment in the network, etc.) according to the parameters p1 to p4 defined by the user. This function must return 1 for success and -1 for error.

```
int get_table_value(int n1, int n2, int n3, int n4,  
t21 *val)
```

Power Assist uses this function to read the value of a variable in the data acquisition equipment. The address of the variable to be read is defined by the user in the parameters n1 to n4. This function must store the value read in the address which pointer is *val*. It will return 1 for success and -1 for error.

```
int refresh_setpoint(int n1, int n2, int n3, int n4,  
t21 *val)
```

Power Assist uses this function to send a value to the data acquisition equipment. The address is defined by the user in the parameters n1 to n4. The value to be sent is stored in *\*val*. This function must return 1 for success and -1 for error.

```
int get_block(int b1, int b2, int b3, int b4, t21  
tab[])
```

Power Assist uses this function to read a block of variables in the data acquisition equipment. The address of the block to be read is defined by the user in the parameters b1 to b4. This function must store the values in the memory area indicated by *tab[]*. It will return 1 for success and -1 for error.

*int stop\_com(int p1, int p2, int p3, int p4)*

This function is called when the user quits Power Assist. It is usually used to restore what had been changed in *start\_com*. It will return 1 for success and -1 for error.

## Driver.h

Besides developing these functions you also have to include the header *driver.h*. This header has the functions that make the I/O driver a TSR (resident program) and set the interruptions.

You can also look at the file *example.c*.

---

## Testing

The easiest way to test the I/O driver is:

a) Loadhigh the I/O driver.

b) Run 21.exe and go to *Edit Tags*.

c) Define:

Number of Drivers	= 1
Number of blocks	= 0
Number of text Tags	= 1
Number of mirror files	= 0
Number of Tags	= 100

d) Define the number of the I/O driver (your ID) and p1 to p4.

e) Ignore text Tags

f) Define TAG0001 as PLC and define n1 to n4.

g) Go to *Insert Setpoints*. Define an area for one tag(one line). Ignore the window of options.

h) Go to *Insert Table*. Define an area for one tag. Ignore the window of options.

i) Go to *Insert Monitor*. The cursor will be over the Setpoint. Type a value and Enter. If the I/O driver is OK the value that you type will appear at the tag in table. If the two values are not matching or if interrogation marks appear there is something wrong with your I/O Driver.

## APPENDIX C - SPC FORMULAS

$n$ : sub-group size  
 $\bar{x}$ : sub-group average  
 $\bar{\bar{x}}$ :  $\bar{x}$  average  
 $\bar{R}$ : range average  
 $\bar{S}$ : standard-deviation average  
 UCL: upper control limit  
 LCL: lower control limit  
 UEL: upper engineering limit  
 LEL: lower engineering limit

$$\text{If } n \leq 10: \quad \begin{aligned} \text{UCL} &= \bar{\bar{x}} + 3 (\bar{R} / (d_2 n^{1/2})) \\ \text{LCL} &= \bar{\bar{x}} - 3 (\bar{R} / (d_2 n^{1/2})) \end{aligned}$$

$$\text{If } n > 10: \quad \begin{aligned} \text{UCL} &= \bar{\bar{x}} + 3 (\bar{S} / (c_2 n^{1/2})) \\ \text{LCL} &= \bar{\bar{x}} - 3 (\bar{S} / (c_2 n^{1/2})) \end{aligned}$$

$$\text{CP} = (\text{UEL} - \text{LEL}) / (\text{UCL} - \text{LCL})$$

$$\text{CPK} = (\text{UEL} - \bar{\bar{x}}) / (\text{UCL} - \bar{\bar{x}}) \text{ or } (\bar{\bar{x}} - \text{LEL}) / (\bar{\bar{x}} - \text{LCL})$$

**TABLE OF CONSTANTS**

<b>n</b>	<b>c<sub>2</sub></b>	<b>c<sub>3</sub></b>	<b>d<sub>2</sub></b>	<b>d<sub>3</sub></b>
2	0,564	0	1,128	0
3	0,723	0	1,693	0
4	0,798	0	2,059	0
5	0,841	0	2,326	0
6	0,869	0,030	2,534	0
7	0,888	0,118	2,704	0,076
8	0,903	0,185	2,847	0,136
9	0,914	0,239	2,970	0,184
10	0,923	0,284	3,078	0,223
11	0,930	0,321	3,173	0,256
12	0,936	0,354	3,258	0,284
13	0,941	0,382	3,336	0,308
14	0,945	0,406	3,407	0,329
15	0,949	0,428	3,472	0,348

## APPENDIX D - OEMS

Power Assist consists of Power Assist Development, Power Assist Runtime and Power Assist Demo.

Power Assist Development is used to develop Man Machine Interface applications. It is also possible to run applications with Power Assist Development, but only for ten minutes.

Power Assist Runtime is used to run Man Machine applications created by Power Assist Development.

With Power Assist Demo you can perform all the functions included in the Power Assist Development package within a limited number of tags (30) and communication for ten minutes only. It does not require a hardkey and can be distributed to potential customers.

As you will notice, Power Assist uses a line in the bottom of the screen where it shows the name of the product (Power Assist). To customize this message you just have to create an ASCII file containing the new message. There are individual files for Power Assist Development, Runtime and Demo, and their names are "TITLE.PAC", "TITLE.PAR", "TITLE.PAD".

Power Assist also uses a VGC file that is used as a background when the program is loaded. To customize it, just create your own VGC file (see *Creating Drawings* in the manual) and name it as "PANEW.VGC".

**This page is blank.**

---

# Index

---

## A

Alarm, 39  
Alarms, 64, 65, 75  
    acknowledge, 67  
    acknowledgment, 76  
    direct alarms ack, 69  
    file size, 65  
    to printer, 66  
Alarms and Events, 64, 75  
Alignment, 50  
Al-low, 41  
Al-up, 41  
Analyzing Data, 76  
And, 37  
Application, 63, 64, 74  
    defining the, 13  
    developing, 18  
    running the, 18  
Automatic refresh, 53  
Avg, 37

---

## B

Background, 46, 48, 50, 53, 54, 55, 57, 58  
Batch  
    end, 68  
    start, 68  
Bbit, 38  
Bit, 38  
Bitmap, 20, 44, 45, 87  
Blocks, 7, 26, 29, 30, 36, 38, 94  
Blocks Window, 29

---

## C

Chart type, 83  
Clear, 13, 60  
Communic, 65  
Const, 37  
Copy, 60

Count, 39  
Counter  
    restart, 70  
Cycle, 32, 33, 47, 62, 71, 73

---

## *D*

DayW, 39  
Dec, 43  
Defaults, 86  
Delete, 60  
Directory, 77, 83  
Display, 51, 55  
DOS, 3, 4, 7, 26, 85  
Drawings, 3, 5, 6, 20, 22, 45  
    creating, 20  
Driver.h, 93  
Drivers, 1, 7, 26, 27, 89, 91  
    configuration window, 27  
Drv, 27, 29, 35, 41

---

## *E*

Events, 64, 67, 69, 76, 86, 87  
    file size, 65  
Expressions, 26, 31  
    window, 31

---

## *F*

Fill, 44, 50, 54  
Fin\_v, 39  
Flash, 37  
    cycle, 54  
Font, 46, 48, 55, 57, 59  
Foreground, 50, 53, 55, 57, 58  
Frame, 46, 48, 55, 56, 59  
Functions, 14, 91  
Functions Window, 66

## *G*

Gp, 42  
Groups, 27, 31, 51, 54  
    read, 70

---

## *H*

Hex, 43  
Historical Analysis, 15, 67, 72, 76, 77, 78, 86, 87  
Histories Window, 71  
History, 47, 72, 76, 86, 87  
    number, 47

---

## *I*

I/O Drivers, 4, 89, 90  
Index, 40  
Information, 62  
Ini\_v, 38  
Installation, 3  
I-Plc, 41

---

## *K*

Keypad, 62, 71  
Keys, 44, 55

---

## *L*

Ln, 39  
Load, 21, 25, 61, 63  
    recipe, 68  
Log, 39

---

## *M*

Menu, 66  
Meter, 58

Mirror, 32, 40, 63, 64, 72, 85  
    write, 70  
Mirror files, 26, 32, 33, 40, 94  
    window, 32  
Mode, 64  
Modify, 60, 61  
Monitor, 43, 61, 62, 63, 94  
Mouse, 3, 4, 56  
Move, 60

---

## *N*

Network, 26, 85  
Not, 38

---

## *O*

Octal, 43  
OEMS, 97  
Options, 5, 20, 62  
Or, 37

---

## *P*

Paintbrush, 5, 20, 21, 22  
Parameters, 27, 29, 41, 90  
Precision, 46, 48, 50, 52, 55  
Print, 43, 67, 69, 73, 76, 78, 85, 86, 87  
    events, 67  
    form, 67  
    history, 67  
    screen, 67  
Printer  
    for alarms, 66  
Printing Colors Window, 73  
Proce, 39  
Procedures Window, 73

## R

Rem, 36  
Retries, 27  
Run, 18, 21, 73, 94

---

## S

Save, 10, 13, 17, 20, 21, 25, 61, 63, 71  
    recipe, 68  
Scale, 46, 48, 49  
    divisions, 47  
Screen background, 5  
Screens, 3, 5, 13, 20, 24, 44, 61, 87  
    change, 69  
    creating, 11  
Setpoints, 21, 45, 52, 53, 54, 75, 85, 94  
    confirm, 68  
    refresh, 70  
    send, 68  
Settings, 24, 62  
Sound warning, 65  
SPC, 67, 76, 77, 83, 95  
    print, 69  
Sqr, 36, 42  
Sub-group size, 83  
Sum, 36

---

## T

Tags, 1, 7, 8, 9, 10, 12, 16, 25, 26, 27, 30, 31, 32, 36, 37, 38, 40, 41, 42, 43,  
46, 47, 48, 49, 50, 51, 52, 53, 54, 63, 64, 65, 68, 69, 70, 71, 72, 75, 76, 77,  
78, 80, 81, 82, 84, 86, 87, 97  
    defining the, 7  
    group of tags window, 31  
    increment, 70  
    jog, 57  
    loads, 25  
    momentary, 56  
    saves, 25  
    set, 69, 85  
    switch, 70  
    text, 75  
    text tags windows, 30

text, 26  
toggle, 56  
verification, 51  
window, 34  
Temp, 37

---

## V

Values  
  number of, 46  
  save, 69  
Verify  
  events, 67  
Visual warning, 65

---

## W

Warning cycle, 65  
Windows .BMP, 5

---

## X

Xor, 38

---

## Z

Zones, 51, 52, 54