



# Dearborn Group *Technology*

## DPA DRIVERS FOR LABVIEW USER'S MANUAL

*Version 1.0*

For the following products:

- DPA II *Plus*
- DPA III (ISA and PC/104 versions)
- DPA III *Plus* (serial versions)
- DPA 4 (USB version)
- DPA RF (wireless version)

© 2004 Dearborn Group Inc.  
27007 Hills Tech Court  
Farmington Hills, MI 48331  
Phone (248) 488-2080 • Fax (248) 488-2082  
<http://www.dgtech.com>

This document is copyrighted by the Dearborn Group Inc. Permission is granted to copy any or all portions of this manual, provided that such copies are for use with the product provided by the Dearborn Group, and that the name “Dearborn Group Inc.” remain on all copies as on the original.

### **IMPORTANT NOTICE**

The DPA drivers for LabVIEW is intended to be used as an evaluation tool only. Damage to the DPA, if caused by misuse, is not covered under the seller’s product warranty.

When using this manual, please remember the following:

- This manual may be changed, in whole or in part, without notice.
- Dearborn Group Inc., does not assume responsibility for any damage resulting from any accident – or for any other reason – while the DPA is in use.
- Examples described herein are for illustration purposes only and do not necessarily represent the latest revisions of hardware or software. Dearborn Group Inc. assumes no responsibility for any intellectual property claims that may result from use of this material.
- No license is granted – by implication or otherwise – for any patents or any other rights of Dearborn Group Inc., or of any third party.

DPA is a trademark of Dearborn Group Inc. LabVIEW is a trademark of National Instruments. Other products are trademarks of their respective manufacturers.

Recent manual revision history:

Jun. 29, 2004      (create initial documentation)

# Table of Contents

<b>1.0</b>	<b>DPA DRIVERS FOR LABVIEW.....</b>	<b>4</b>
1.1	Included files .....	5
1.2	DPALVInitDPA.vi .....	6
1.3	DPALVRestoreDPA.vi .....	7
1.4	DPALVCheckDataLink.vi .....	8
1.5	DPALVInitDataLink.vi.....	9
1.6	DPALVLoadMailBox.vi .....	10
1.7	DPALVUnloadMailBox.vi.....	11
1.8	DPALVReceiveMailBox.vi .....	12
<b>I</b>	<b>INDEX .....</b>	<b>13</b>

## 1.0 DPA Drivers for LabVIEW

DPA Drivers for LabVIEW is a collection of LabVIEW VI device drivers used to create VIs that interact with the DPA family of products. They rely on drivers installed from the DPA Family Installation CD-ROM.

The DPA was designed to work with applications written in C/C++ and uses structures for passing in most of its data. As much as possible the DPA driver VIs duplicate C functions, and the clusters they use duplicate the C structures. This manual assumes the reader is competent with LabVIEW and DPA programming.

The DPA drivers for LabVIEW consist of three main parts. The first part is the actual VIs used in LabVIEW (contained in `DPALV.11b`). The second part is a wrapper or “helper” DLL that converts data from LabVIEW types to those used by the regular DPA drivers (`DPALV32.dll`). The third part is the regular DPA native drivers (obtained from the DPA installation CD-ROM).

As much as possible, the VIs correspond directly to functions in the regular DPA native drivers, although some inputs have been simplified or left out to reduce complexity. The LabVIEW VIs are named using the convention `DPALV<driver function name>.vi`. More information about each function can be found under that function in the DPA Manual. This document will often omit the “DPALV” and the “.vi” from the names for convenience and ease of reading.

The included examples use a cluster on the control panel to create the clusters used as inputs to the DPA SubVIs. You may find it easier to use separate controls and a `Bundle` or `Bundle by Name` function to create the clusters, especially if some or all of the elements are constants instead of taken from the control panel. An `Unbundle` or `Unbundle by Name` could be used to extract individual elements from the output cluster of `ReceiveMailBox`.

To begin, you must always call `InitDPA` to open a port to the DPA and get a handle to it. This handle is then passed to all other DPA functions. It is passed back out of all of them as well, so that you can use it to control data flow and thus execution order. At the end of the session you must call `RestoreDPA` to unload the drivers and release the port.

## 1.1 Included files

The following files are part of this *DPA Drivers for LabVIEW* release.

LabVIEW_DPA.pdf	;this document
DPALV.llb	;LabVIEW library, a collection of separate VIs
DPALV32.dll	;LabVIEW -specific driver, converts LabVIEW types to DPA driver types

Also included are example VIs using different methods to display return status information. Each contains indicators that you can copy and paste.

ReturnStatusType_cluster.vi	;Display return status (method #1)
ReturnStatusType_display.vi	;Display return status (method #2)
ReturnStatusType_enum.vi	;Display return status (method #3)

The following examples correspond to DPA functions.

```
test_LVCheckDataLink.vi
test_LVInitDataLink.vi
test_LVLoadMailBox.vi
test_LVReceiveMailBox.vi
test_LVUnloadMailBox.vi
```

### Notes:

The following notes reflect the most current information about this release.

- These files may be available on CD-ROM or as a downloaded “.ZIP” file.
- To avoid LabVIEW “file not found” problems, put the **DPALV.llb** and the **DPALV32.dll** file in the same directory as the native drivers. Then set up your VIs to use them from that directory.

## 1.2 DPALVInitDPA.vi

### Description:

Open port to DPA.

### Inputs:

Device ID

### Outputs:

status  
DPA handle

### Notes:

Use **DPALVRestoreDPA.vi** to release the DPA before exiting. Returns status and handle to DPA. This handle must be passed to other DPA VIs. The Device ID corresponds to an entry in the **DG\_DPA32.INI** file, usually located in the Windows directory.

## 1.3 DPALVRestoreDPA.vi

### Description:

Release DPA port.

### Inputs:

DPA handle

### Outputs:

status

### Notes:

Must be called to release resources used by `DPALVInitDPA.vi`.

## 1.4 DPALVCheckDataLink.vi

### Description:

Get version info from DPA.

### Inputs:

DPA handle

### Outputs:

status

DPA handle

version string

### Notes:

String format is:

Dearborn,<driver ver>,<hardware ver>,<firmware ver>,<protocols supported>,<scratchpad buffer size>,<serial baud rate>

## 1.5 DPALVInitDataLink.vi

### Description:

Initialize protocol.

### Inputs:

DPA handle  
a cluster that defines the protocol

### Outputs:

status  
DPA handle

### Notes:

`InitDataLink` is called once per protocol to configure protocol-specific items such as baud rate, message format, hardware configuration, etc. Specific values for the parameters in the cluster depend on the protocol being initialized. See the DPA Manual for more information.

## 1.6 DPALVLoadMailBox.vi

### Description:

Load a DPA MailBox.

### Inputs:

DPA handle  
a cluster defining the MailBox  
a refnum for an occurrence

### Outputs:

status  
MailBox handle. This handle must be passed to other MailBox VIs.

### Notes:

The occurrence refnum can be used as a callback. Each time the mailbox transmits or receives a message, the occurrence is set (triggered). Part of the diagram can be made to wait for the occurrence, and then perform some action, such as increment a counter or retrieve and display the message data. To execute that part of the diagram once per message it must be placed in a loop.

Not all items in the cluster are used, depending on whether the MailBox is Transmit or Receive, and on which protocol it is for. See the DPA Manual for specific definitions of each parameter.

## **1.7      DPALVUnloadMailBox.vi**

### **Description:**

Unload a DPA MailBox.

### **Inputs:**

DPA handle  
handle of MailBox to be unloaded

### **Outputs:**

status

### **Notes:**

Unloads a mailbox previously loaded by `DPALVLoadMailBox.vi`.

## 1.8 DPALVReceiveMailBox.vi

### Description:

Get data from a Receive MailBox.

### Inputs:

DPA handle  
MailBox handle

### Outputs:

status  
a cluster with data

### Notes:

`ReceiveMailBox` will return the most recent message received by that MailBox. If it's called a second time before another message comes in it will return the first one again. So it should generally be called from a block diagram loop that's waiting on an occurrence from the mailbox. This ensures that there is data to be received, and that each message is received only once.

Not all items in the cluster are meaningful, depending on the protocol. See the DPA Manual for more information on specific items.

## INDEX



## I Index

DPALVCheckDataLink.vi, 8  
DPALVInitDataLink.vi, 9  
DPALVInitDPA.vi, 6  
DPALVLoadMailBox.vi, 10

DPALVReceiveMailBox.vi, 12  
DPALVRestoreDPA.vi, 7  
DPALVUnloadMailBox.vi, 11