



FLYBOX[®]
HARDWARE MANUAL

Contents

CHAPTER 1 OVERVIEW	5
1.1 WHAT'S NEW	5
1.2 CONVENTIONS USED IN THIS MANUAL	6
CHAPTER 2 INSTALLATION	7
2.1 CONNECTING TO POWER	7
2.2 CONNECTING TO THE HOST WORKSTATION	9
2.3 ADDING ADDITIONAL INPUTS	9
CHAPTER 3 HARDWARE	11
3.1 THE JOYSTICK	11
3.2 LEVERS	12
3.3 SWITCHES	12
CHAPTER 4 SOFTWARE	13
4.1 SOME BASICS	13
4.1.1 Connecting the FlyBox	13
4.1.2 Theory of Operation	13
4.2 TUTORIAL FLYBOX_TEST.C	14
4.2.1 Setup	14
4.2.2 Opening and Initializing	15
4.2.3 The Interval Timer	16
4.2.4 Cycling	16
4.3 SOFTWARE MAPPING	17
4.3.1 Optional Digital Inputs	17
4.3.2 Testing the Digital Inputs	17

CHAPTER 5 TROUBLESHOOTING	19
5.1 THE FLYBOX IS NOT RESPONDING	19
5.1.1 Serial Cable	19
5.1.2 No Power	20
5.1.3 Joystick Problems	20
5.1.4 Interference	20
CHAPTER 6 TECHNICAL SPECIFICATIONS	21

CHAPTER 1 OVERVIEW

The FlyBox® is an input device for your computer. It consists of a three axis joy stick (the third axis is twist), two levers, and several discrete switches. It provides inputs to the workstation through an RS-232 serial line, and can be used instead of the mouse or a dial and button box to move through a terrain database.

It provides a superior feel for dynamic applications. It can be used for flight or driving simulation, database construction, etc. The FlyBox is packaged in an aluminium case, and has a simple DB-9 or MiniDIN-8 cable connection to the computer.

It provides the following advantages over systems that use an A/D board:

- *You don't have to rebuild the UNIX kernel to include a special driver.*
- *You don't have to open the computer to install an Analog to Digital board since the A/D conversion is handled internally by the box.*
- *The software interface is extremely simple, using standard, portable open, and read calls.*
- *The FlyBox can update as fast as 50Hz, which is generally faster than inputs can be generated.*

This manual contains instructions for installation of the FlyBox, some highlights of the software used to communicate with the FlyBox, and some troubleshooting hints. This manual should be read in conjunction with the **LV824 Software Manual**, which contains specific details about the software that is used on the host to communicate with the LV824 circuit board inside the FlyBox.

The software section of this manual is written in tutorial style, stepping through some example software. If you have problems, please consult Chapter 5 of this manual - Troubleshooting. Most problems in getting started are related to the workstation configuration.

1.1 WHAT'S NEW

The old **FlyBox Owner's Guide** has been split into this **FlyBox Hardware Manual** and the companion **LV824 Software Manual**. This allows us to update the software on the LV824 without having to revise all the hardware manuals for the products that use the LV824. Reference is made in this manual to software features that are specific to the FlyBox, but the Software Manual is intended to be the primary reference.

The FlyBox now uses the JFx joystick which allows a number of optional switch configurations. These are discussed in section 3.1.

The page numbers in this manual have been modified so that links in the "on-line" version are accurate. (This just means that page 1 is the front cover, and roman numerals are no longer used for the table of contents). Documentation can be down-loaded from the BG Systems web site at any time.

1.2 CONVENTIONS USED IN THIS MANUAL

Various typefaces are used in this manual to refer to the name of something on the computer, to highlight an important piece of information, etc.

Italics are used in the body of the text to indicate computer terminology - typically a function or file name:

open_lv()

A monospace font is used whenever a code fragment is presented:

```
tios.c_flag = CS8 | CREAD | CLOCAL;
```

A monospace font preceded by a % is used for Unix shell commands:

```
% cd FlyBox/
```

When the font is preceded by a #, it means that system administrator (or root) privileges are required for the operation

```
# chmod a+rw /dev/ttyd2
```

Things that are really important are set apart with a “hand” pointing to them, and an italic font is used:

 *WARNING.*

Things that are should be noted, but are not dangerous if ignored, are set apart with the spectacles:

 *NOTE.*

CHAPTER 2 INSTALLATION

The first step is to verify that you received all the necessary parts. As you unpack the box make sure that everything on the packing list is supplied. If anything is missing contact BG Systems immediately.

The FlyBox is quite solidly constructed, but you should ALWAYS carry it by the handles — NEVER pick the FlyBox up by the joystick, this can cause damage to the stick, and puts stress on the case.

2.1 CONNECTING TO POWER

The power switch is located on the right hand side as you look at the back of the case. The FlyBox is set for 110v when shipped within the U. S. A. International shipments are usually set to 220v. Before connecting the power cord, make sure that the voltage is correctly set.

If the voltage selector is incorrectly set, or you are taking your FlyBox to a trade show where the power requirements are different, the procedure for changing the input voltage is as follows:

- 1 Remove the fuse drawer (see figure 1), by using a screwdriver to push at the drawer release (see figure 2).

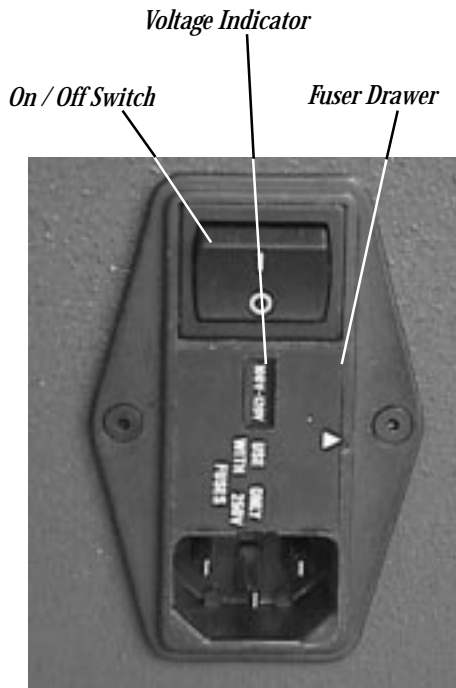


Figure 1 Power Connector

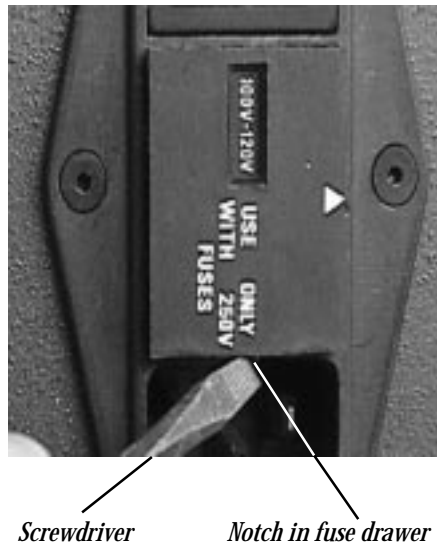
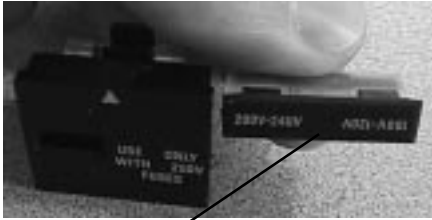


Figure 2 Removal of fuse drawer



Rotate this through 180° and slide back in

Figure 3 Detail of voltage selector

- 2 Slide out the voltage selector, the T shaped board, as shown in figure 3.*
- 3 Turn it through 180° and slide it back in the channel so that the desired voltage can be seen through the window.*
- 4 Making sure that the white triangle on the fuse drawer is aligned with the triangle on power connector, push the fuse drawer back into place.*

If power is correctly connected, the red LED on the front of the FlyBox will light up. Also, at the back of the FlyBox next to the serial connector there are two LEDs. The green LED comes on with power to the LV824 inside the FlyBox, and the yellow LED shows status for the LV824. On power up the yellow LED will blink once.

🔍 If the red LED on the front of the FlyBox does not come on, it is possible that the fuses have been blown. Remove the fuse holder as described above and inspect the fuses. If they have “blown” there are 2 spare fuses in the fuse drawer as shown in figure 4. Just push out the two rectangular boxes underneath the fuses.

☞ The power cord that comes with the FlyBox has a THREE pin connector. DO NOT CUT OFF THE THIRD PIN. This provides ground for the system. If you do not have a grounded circuit, contact an electrician. Damage to the FlyBox and the workstation can result from an ungrounded system.

☞ If the FlyBox is being used in a foreign country where the voltage is 220v, it is not uncommon for users to forget to switch the voltage until AFTER they have blown the fuse. Please use the spare 500mAmp SloBlo fuse (figure 4) rather than putting in a 2 amp fuse. Damage to the power supply will result if connected to 220v while set to 110v.

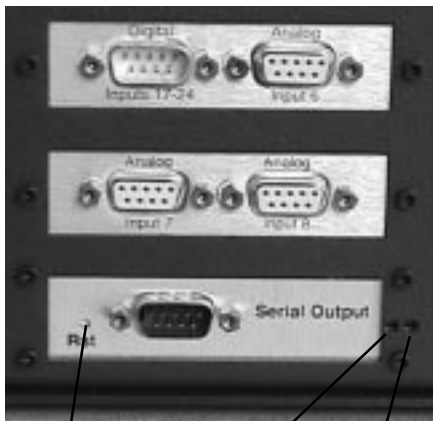
☞ If the FlyBox is set to 220v and connected to 110v, it may appear to work for a while, but when you push one of the illuminated switches it stops communicating. This is because the voltage to the LV824 drops below 5 vDC. Changing the voltage selector back to 110v will fix this problem.



Figure 4 Location of spare fuse

2.2 CONNECTING TO THE HOST WORKSTATION

The FlyBox should be set up at a convenient location near the monitor of the computer. Make sure that the power switch on the back of the FlyBox is in the off position, and connect the power cable. Then connect the RS-232 cable to the FlyBox serial output port (see figure 5) and the host computer. You can connect the FlyBox to any available “port” on the back of the host computer. Note that the female end of the cable connects to the FlyBox and the male end of the cable connects to the host. The pins on the DB-9 are: pin 2 - RX, pin 3 -Tx and pins 5 and 7 - Ground. Pinouts for the various different serial cables are shown in section 4.1.1.



Reset button

Status LED

Power LED

Figure 5 Back panel detail

2.3 ADDING ADDITIONAL INPUTS

The A/D board inside the FlyBox can handle up to 8 analog and 24 discrete inputs. The standard configuration of the box uses 5 analog and 9 discrete inputs. Consequently we have provided 4 extra DB-9 ports on the back of the FlyBox, see figure 5.

Additional discrete inputs can be connected to the port labelled Discrete Inputs 17-24. A female DB-9 connector should be used, and the pins should be wired as follows:

Pin	Channel
1	17
2	18
3	19
4	20
5	21
6	22
7	23
8	24
9	ground

Note that the voltages must be 0v in the off state, and +5v in the on state. If you have a FlyBox that uses more than 16 discrete inputs (i.e. with two coolie hat switches), this input DB-9 will not be connected to the LV824.

Additional analog inputs can be connected to the DB-9's labelled Analog Input 6, 7, and 8. These use a male DB-9 connector, with pin 1 connected to the input voltage, and pin 9 connected to ground. The voltage range must be between 0-5v. Pin 8 provides 5v output which can be used to power the device that you are connecting to the FlyBox. Note, however that you should not try to draw more than 200 mAmps from the FlyBox.

CHAPTER 3 HARDWARE

The FlyBox consists of three major control groups. The joystick on the right hand side, the levers on the left hand side, and the pushbuttons in the center. There are a number of customized versions (e.g. the Unit Training Device FlyBox) for which some of the details will differ from those described in this chapter.

The FlyBox is now available with a two axis joystick, and there are a wide variety of options that can be added to the JFx joystick.

3.1 THE JOYSTICK

The standard FlyBox (BG-530) has a three axis joystick with the motion in the following directions: Forward/Backward (pitch), Left/Right (roll) and Twist (yaw). There is a trigger on the joystick which is a digital input. The three axes of the joystick provide a proportional output between 0-5 volts, which is converted to a 12 bit analog signal scaled between -1.0 and 1.0.

The FlyBox can also be ordered with a two axis joystick (BG-520) in which case there is no twist capability. A proportional thumb rocker can be added to either 2 or 3 axis joystick to provide an additional analog input.

At the head of the joystick it is also possible to mount a variety of momentary, latching, 2-position and 4-position switches. Figure 6 shows a common configuration with two momentary switches and a 4-position switch. See our current price list for the various JFx options.

The mapping between the physical controls and the software structure is described in chapter 4.



Figure 6 JF3-1-MP-CH-MP option

3.2 LEVERS


Two levers are mounted on the left side of the FlyBox. In the standard FlyBox configuration these are in a friction hold mode. The levers can be ordered with a spring centering mechanism if required.

They produce a voltage which is fed to an amplifier circuit board which is calibrated to ensure that the output voltage covers the full 0.0 to 5.0 volt range. This voltage is then converted to an digital value by the LV824 circuit board, and the software values run from -1.0 to 1.0.

3.3 SWITCHES

There are 6 discrete switches (red covers) mounted on the FlyBox, nearest the top. These switches produce a discrete (on/off) reading. They are on when pushed in, and when on they are lit. The discrete switches are spring loaded and stay down until pressed again.

There are two momentary switches (orange covers) mounted at the bottom of the cover. These are on ONLY when pressed, and off when released.


 *This is the standard switch configuration. If you need more momentary switches, you can order FlyBoxes with your specifications for switch combinations.*

CHAPTER 4 SOFTWARE

This chapter describes the software interface to FlyBox. This chapter should be read in conjunction with the more detailed descriptions in the **LV824 Software Manual**. The primary purpose of this chapter is to indicate the relationship between the physical aspects of the FlyBox and the software data structure.

To get the most out of our products a good understanding of your application is also required, since the demands of real-time simulation, low frequency data acquisition and model viewing are so different.

We provide source code for an example program that interface to the FlyBox, which you should be able to modify to suit your needs.

 *The software provided may be copied and used as needed. The library functions should be changed with caution. The examples are intended to act as templates for your own software applications.*

In fact you may be able to get away without in depth knowledge, but if you do encounter problems, please read this chapter thoroughly.

Section 4.1 reviews the basics of connecting the FlyBox to the host computer. Section 4.2 is a tutorial which studies an application that communicates with the FlyBox.

4.1 SOME BASICS

4.1.1 Connecting the FlyBox

Once you have connected the FlyBox to the host with an appropriate serial cable and connected power, you are ready to test the connection to see if you can communicate with the FlyBox. However, there are some parameters that you need to be aware of on the host in order to get started. (Note that this is covered in full in section 2.2 of the **LV824 Software Manual**.)

The first thing to do is decide which serial port to connect the FlyBox. The serial ports are typically numbered 1-n, with 1 typically reserved for use as an alternate console. Therefore we recommend that you choose port 2 or higher.

Once you have selected a serial port, we recommend that you use the Unix environment variable feature:

```
% setenv FBPORT /dev/ttyd2
```

When the BG software tries to open the serial port, the first thing that it looks for is this environment variable, and once things are working, you should add this line to your *.cshrc* file so that it is set every time you log in.

4.1.2 Theory of Operation

It is also useful to understand the basic sequence of events that are required to use the FlyBox. Since the FlyBox is an input device, we assume that the host needs to collect the inputs at some regular interval. Some applications will need updated information faster than others, but it is important to understand that the data transfer is being made via RS-232 serial protocol, and there are finite amounts of time involved. The steps required are:

1. Send character to FlyBox requesting data
2. Wait for data to arrive
3. Read data from the port

Typically you would use the time waiting in step 2 to perform whatever processing is required based on the

last set of data received. It is important that you wait long enough for the data to arrive and this will be determined by the baud rate and the number of channels that you are sampling. It may well be that the processing takes longer than the time required to wait, but if you don't have much processing to do, then you will have to include some sort of timed wait in order to avoid looking for data that hasn't arrived.

4.2 TUTORIAL FLYBOX_TEST.C

Perhaps the easiest way to understand how to use the software is to study an example. In the *bg/FlyBox* directory, the source code in *flybox_test.c* is a fairly simple program that samples the FlyBox and prints the values to the terminal.

There is a *Makefile* in the directory, and you should be able to type:

```
% make flybox
```

to compile and link the executable *flybox*.

4.2.1 Setup

As you go through this example, look at the file *flybox_test.c* and make sure that it matches this document.

The low level library routines use a data structure, defined as *bglv*, that is defined in *lv3.h*. (Note that this file is in the *bg* directory.) This data structure must be declared in the main file.

```
#include "lv3.h"  
bglv bgdata;  
// Main BG data structure
```

Note the use of C++ style comments // which are not in the actual code, but are used here to help readability.

Only the following members need to be set.

```
int    analog_in;  
// Analog input selector  
int    dig_in;  
// Digital input selector  
int    baud;  
// Baud rate selected
```

*The remaining members of the structure are computed in *open_lv()* and *init_lv()*.*

The *bglv* structure is passed to most of the library functions in the call sequence. In order to collect data from the FlyBox, the serial port needs to be opened by


your program, and the appropriate configuration sent to the FlyBox. The function *setup_lv()* in the example performs this task.

```
bgdata.analog_in = 0;
bgdata.analog_in = AIC1 | AIC2 | AIC3
                  | AIC4 | AIC5;
```

The *analog_in* member of the *bgdata* structure sets any combination of the 8 available analog inputs. In the example above, channels 1, 2, 3, 4, and 5 have been requested by “or-ing” the predefined values *AIC1 - AIC5*. If you have connected an additional analog input to the DB-9 at the back of the FlyBox, you could add it to the list:

```
bgdata.analog_in |= AIC8;
// Analog input 8
```

If you examine the definitions in *lv3.h*, you can of course use the shorthand hex representation to set these parameters.

 *Note that we use the logical (not C based) counting system that assigns AIC1 to pin 1 on the connector. Of course in C this corresponds to the zero element in the array.*

Next we select the digital (discrete) channels:

```
bgdata.dig_in = 0;
bgdata.dig_in = DIC1 | DIC2;
```

In the example above, we have selected digital inputs 1-16. These are selected according to the following table:

DIC1	0x10	Channels 1-8
DIC2	0x20	Channels 9-16
DIC3	0x40	Channels 17-24

These can be combined in any desired way. For example:

```
bgdata.dig_in = DIC1 | DIC3;
// channels 1-8 and 17-24.
```

4.2.2 Opening and Initializing


First we set the baud rate in for the *bgdata* structure:


```
bgdata.baud = BAUD192;
```

The next step is to open the serial device on the host with the *open_lv()* function. The call takes three arguments: the first is the address of the BG data structure the second specifies the */dev/tty* that you are connected to; and the third specifies whether the read will be blocking or non-blocking.

```
st = open_lv(&bgdata, "/dev/ttyd2",
            FB_NOBLOCK);
```

In general for SGI workstations, setting the baud to 19200 as shown above should work well, and the default on power up for the FlyBox is 19200. This is important when you try to send initialization data to the FlyBox, since if your serial port is configured for a different baud rate, nothing will get through! The *open_lv()* function sets the baud rate to 19200 to begin with, and the *init_lv()* function sets the baud rate to the desired operational rate.

 *Note that when you push the reset button or cycle power, the LV824 reverts to 19200 baud.*

 *Details of the open_lv() function can be found in the LV824 Software Manual*

During the *open_lv()* call, the software sends a character to the FlyBox to retrieve the EPROM version. This is stored in the *bgdata.Rev* structure:

```
bgdata.Rev.year
// Year the EPROM s/w was written
bgdata.Rev.major
// Major release (currently 3)
bgdata.Rev.minor
// Minor release (currently 0)
bgdata.Rev.bug
// Bug release (currently 6)
bgdata.Rev.alpha
// EPROM version e, f, or g.
```

This data is available for checking by your application, and is used in the *init_lv()* routine to make sure that the requested setup matches the hardware.

The information from the revision and the channel setup is used in the call to initialize the FlyBox:

```
st = init_lv(&bgdata);
```

This call simply sends a string to the FlyBox that tells it what baud rate to run at, and which channels to sample.

Note that the FlyBox starts out assuming communication at 19200 baud. If you want to operate at a different baud rate, you can set this when you `init_lv()` with `bgdata.baud` set to `BAUD9600` for example. However, be aware that if you don't close `lv()`, the FlyBox will stay at 9600 baud, and the next time you initialize, the changes will not be received. You need to either push the reset button or turn power off and on to reset the FlyBox to 19200.

The `init_lv()` routine performs some compatibility checks. For example, if you have selected:

```
bgdata.dig_out = 0x70;
```

then the `init_lv()` call will fail because you cannot use the digital output feature on an LV824-E.

4.2.3 The Interval Timer

Next we call the function `init_timer()` which sets up the interval timer, to trigger an alarm every 50,000 microseconds (50 ms).

```
itv.it_interval.tv_sec = 0;
itv.it_interval.tv_usec = 50000;
itv.it_value = itv.it_interval;
setitimer(ITIMER_REAL, &itv, (struct
itimerval *)0 );
```

Next we set up to catch the alarm signal sent by the interval timer. (The routine that we call does nothing.)

```
sigset(SIGALRM, catcher);
```

Using the interval timers is simply a method that we have found to be reliable. On high end Silicon Graphics machines with multiple processors, you may be able to get more accurate or fine grained timing by using the graphics subsystem clock. You might look into the Performer software system calls to see how to get a more accurate clock.

See also section 3.3 and 4.1 of the LV824 Software Manual for a more detailed discussion of sampling rates.

4.2.4 Cycling

The normal cycle looks something like this:

```
st = w_lv(bgdata.sp_fd, "o");
// request data
sigpause(SIGALRM);
// wait for the alarm
st = r_lv(&bgdata);
// get the data
```

After reading the data, we choose to print it to the terminal in this example. This is of course where you would choose to do something interesting with the data.

The input values are computed within the `r_lv()` call, and stored in the `bgdata.ain[]` array and the `bgdata.din[]` array. The analog values are scaled between -1.0 and 1.0 for convenience. The digital values are packed into the integers in the array, so you need to mask (as in the example) to determine which bits (channels) are set.

If you want to run this as a single process, you would usually put the intensive processing just before the `sigpause()`. The `sigpause()` suspends your process until the signal is caught, so you should do your processing before you are suspended.

4.3 SOFTWARE MAPPING

The physical controls on the FlyBox are mapped into the `bgld` data structure according to the following tables.

The analog values are stored as floats between -1.0 and 1.0 and each analog channel is represented as a separate member of the data structure. The buttons, switches and trigger are all digital inputs, which are packed together in groups of 8.

Table 1 Analog Inputs

<code>bgdata</code>	Function	Negative	Positive	Note
<code>ain[0]</code>	JFx lateral	Left	Right	
<code>ain[1]</code>	JFx longitudinal	Back	Forward	
<code>ain[2]</code>	JFx twist	Left	Right	Not on JF2
<code>ain[3]</code>	Left Lever	Back	Forward	
<code>ain[4]</code>	Right Lever	Back	Forward	
<code>ain[5]</code>	Thumb Rocker	Left	Right	Optional

Table 2 Digital Inputs

<code>bgdata</code>	Function	Bit	Shift	Note
<code>din[0]</code>	Top Left	0x01	0	Latching
<code>din[0]</code>	Top Right	0x02	1	Latching
<code>din[0]</code>	Mid Top Left	0x04	2	Latching
<code>din[0]</code>	Mid Top Right	0x08	3	Latching
<code>din[0]</code>	Mid Low Left	0x10	4	Latching
<code>din[0]</code>	Mid Low Right	0x20	5	Latching
<code>din[0]</code>	Low Left	0x40	6	Momentary
<code>din[0]</code>	Low Right	0x80	7	Momentary
<code>din[1]</code>	Trigger (1)	0x01	0	

4.3.1 Optional Digital Inputs

The following table presents the mapping for the optional switches. Of course there are some combinations that are not available, and also some of the mappings occur more than once.

Table 3 Optional Digital Inputs

<code>bgdata</code>	Function	Or with	Shift	Note
<code>din[1]</code>	4-pos left	0x02	1	“Coolie Hat”
<code>din[1]</code>	4-pos right	0x04	2	
<code>din[1]</code>	4-pos down	0x08	3	
<code>din[1]</code>	4-pos up	0x10	4	
<code>din[1]</code>	Left button	0x20	5	
<code>din[1]</code>	Right button	0x40	6	
<code>din[1]</code>	Trigger (2)	0x80	7	2-pos trigger
<code>din[1]</code>	2-pos left	0x02	1	
<code>din[1]</code>	2-pos right	0x04	2	
<code>din[1]</code>	2-pos down	0x02	1	
<code>din[1]</code>	2-pos up	0x04	2	

Note that the two position switch can either be a left/right or an up/down.

4.3.2 Testing the Digital Inputs

To determine the state of a given digital input you need to mask it with the appropriate **bit** or **shift** and mask with `0x01`. For example, the trigger is always digital input number 9, which is mapped as the first bit of the second member of `bgdata.din[]`. So to see if the trigger is on:

```
if ( (bgdata.din[1] >> 0) & 0x01 )
    printf("trigger on\n");
```

Note that we shift the value to the right by the physical number of the switch minus one. So, to test the state of the top right latching pushbutton, number 2 of input block 1:

```
if ( (bgdata.din[0] >> 1) & 0x01 )
    printf("top right on\n");
```

Alternatively we can test directly with the appropriate bit

```
if ( bgdata.din[0] & 0x02 )
    printf("top right on\n");
```

but this doesn't work as well when we put this code fragment into a loop.

CHAPTER 5 TROUBLESHOOTING

Every FlyBox is tested prior to shipping, and we have confidence that when you connect a FlyBox it will work. However, since we have two pieces of hardware (host and FlyBox), three pieces of software (host, FlyBox, and the operating system), and a cable, we realize that there will be occasions when you connect a FlyBox it will appear that nothing is happening.

Chapter 7 of the LV824 Software Manual also contains trouble shooting advice, and should be checked for information regarding host setup.

5.1 THE FLYBOX IS NOT RESPONDING

You have plugged everything in, and you try to run a test program (*eprom* or *flybox*), and the program exits. Check the following:

1. *Is the red power LED on the front of the FlyBox on? If not check the fuses and power connection.*
2. *Is the green LED at the back of the FlyBox on? If not check your power supply. If the red LED is on, and the green LED is not, contact BG Systems.*
3. *Does the yellow LED blink when you turn on power? If not then the FlyBox is not working. Call BG Systems for help.*
4. *If the yellow LED blinks when you apply power, it should also blink when you run a test program. If it does not, then the character from the Unix host has not made it to the FlyBox. Check your cables and the host configuration.*
5. *Did the test program print any helpful text on the screen? If you do not have permission to write to the serial port, the software should tell you. In this case contact your system administrator for help.*

A common software problem is that the host is configured with the requested serial port as a terminal running at 9600 baud. The FlyBox software may run for a short period (with the baud at 19200), but Unix will come along and reset the baud rate to 9600 - at which point the FlyBox will stop responding.


5.1.1 Serial Cable

The most common hardware problem is when there is a problem with the serial cable. This problem is compounded by the fact that SGI uses three different types of serial connector (although it appears they have finally settled on the PC DB-9). We make every effort to

ship the correct serial cable with the products, but if you move the FlyBox to a new workstation for which you do not have a BG serial cable proceed with caution. The tables below list the functions on the pins for the various connectors:

Signal	SG-DB-9	MiniDIN-8	PC-DB-9
Rx	3	3	2
Tx	2	5	3
Gnd	7	4	5

Signal	LV824
Rx	2
Tx	3
Gnd	5 and 7

 ***Make sure to connect Rx on the LV824 to Tx on the serial cable !***

If the LV824 stops responding when you connect to a new SGI workstation, please consider the serial cable as a likely problem. The quickest test of the BG hardware is to reconnect it to the last known working machine. If it works with one computer, but not with another, then the BG hardware is OK, and you should double check the serial cable.

5.1.2 No Power

As described in section 2.1 there are circumstances under which the FlyBox will stop operating due to blown fuses or incorrect voltage selection. One of the more interesting problems occurs when the voltage selector is set to 220v and the power provided is only 110v. Under this condition, the FlyBox will often work for a while, but when a load is placed on the power supply (i.e. when a light bulb is turned on), the DC voltage to the LV824 drops below 5, and it stops functioning. This gives the appearance of a FlyBox that is acting “flakey”. Please double check the voltage selector.

5.1.3 Joystick Problems

The new JFx joysticks are built from machined aluminium and stainless steel and use sealed ball

bearings on the gimbals for high precision operation. As with all mechanical assemblies, it is impossible to completely eliminate the dead band at the joystick center position. The JFx is built such that this deadband should be within +/- 0.01 on the *x/y* test program. This represents a 1% tolerance. Older FlyBoxes used a joystick with a 3% tolerance.

5.1.4 Interference

The FlyBox will not electromagnetically interfere with the operation of the host computer. However, there are some peripheral devices which will either cause interference with the FlyBox, or which will not operate in close proximity to the FlyBox.

Since the FlyBox contains a power supply, some magnetic sensors will be affected when they are in close proximity to the FlyBox. This can occur with certain tracking devices. The sensors on the JFx joystick are Hall Effect magnetic sensors, and so operating the FlyBox in close proximity to a strong electromagnetic producer will cause problems.

We do not recommend using the FlyBox in such circumstances. If you have a requirement to use a FlyBox under these conditions, please contact BG Systems for advice.

CHAPTER 6 TECHNICAL SPECIFICATIONS

The following section outlines the various technical specifications for the FlyBox:

A/D Resolution	12 bits
Baud Rates Supported	2400, 4800, 9600, 19200, 38400, 57600, 115200
Power Supply	Switchable between 110vAC and 220vAC at 50/60 Hz. 750 mAmp capability.
Power Draw (LV824-E)	240 mAmp
Serial Interface	RS-232 (Rx, Tx, Gnd)
Hardware Handshake	No
Dimensions	14" x 11" x 5" for the case. JFx adds 7" to the height.
Weight	9.25 lbs (4.2 Kg)
Lights	Backlit switches need a 40 Amp, 6 vDC bulb
Fuses	500 mAmp SloBlo





BG Systems, Inc. Palo Alto, California, USA

BG SYSTEMS, INC.

Tel: 650-858-2628

Fax: 650-858-2685

URL: www.bgsystems.com