



CEREALBOX[®]

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
CHAPTER 3 SOFTWARE	9
3.1 SOME BASICS	9
3.1.1 Connecting the CerealBox	9
3.1.2 Theory of Operation	9
3.2 TUTORIAL E_TEST.C	10
3.2.1 Setup	10
3.2.2 Opening and Initializing	11
3.2.3 The Interval Timer	12
3.2.4 Cycling	12
3.3 F_TEST.C FOR THE IV824-F	13
3.3.1 Setup	13
3.3.2 Cycling	13
3.3.3 Testing	13
3.4 G_TEST.C FOR THE IV824-G	14
3.4.1 Setup	14
3.4.2 Cycling	14
3.4.3 Loop-back Test	14
3.4.4 h_test.c for IV824-H	14
3.4.5 Encoder Counters	14
4 TROUBLESHOOTING	15
4.1 THE CEREALBOX IS NOT RESPONDING	15
4.1.1 Serial Cable	15
5 TECHNICAL SPECIFICATIONS	17

CHAPTER 1 OVERVIEW

The CerealBox[®] is a small device that takes electrical input signals and sends them to a computer through the serial port. There are currently 4 versions of the CerealBox with the following characteristics:

LV824-E	8 analog inputs, 24 digital inputs.
LV824-F	8 analog inputs, 24 digital inputs/outputs.
LV824-G	8 analog in, 24 digital in/out, 3 analog outputs.
LV824-H	8 analog in, 24 digital in/out, 8 analog outputs.

CerealBoxes can also be provided with either 4 or 8 encoder counters, and this is specified by either a -4e or a -8e following the part number.

The CerealBox is intended for use where update rates in the 30-100 Hz range are required. Since it relies on serial communication, the bandwidth is limited by the baud rate supported on the host computer. The CerealBox can be operated at baud rates of up to 115200. Typical use is at 19200 baud.

This manual contains instructions for installation of the CerealBox, some highlights of the software used to communicate with the CerealBox, and some troubleshooting hints. It is intended that this manual be used in conjunction with the **LV824 Software Manual**, which contains specific details about the software that is used on the host to communicate with the circuit board (LV824) inside the CerealBox. The software chapter of this manual is intended for users who are already familiar with the BG software interface and just need pointers for the specific version of the CerealBox that they are using.

The software section of this manual is written in tutorial style, stepping through some example software. If you have problems, please consult Chapter 4 of this manual - Troubleshooting. Most problems in getting started are related to the workstation configuration.

1.1 WHAT'S NEW

The old **CerealBox Owner's Guide** has been split into this **CerealBox 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 CerealBox, but the Software Manual is intended to be the primary reference.

The manual has been updated to cover the use of the encoder counters.

The page numbers have been changed to support the "pdf" version of the manual, so that page 1 is now the cover page.

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 CerealBox/
```

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 “lightning bolt” next 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 information bullet:

 ***NOTE.***

CHAPTER 2 INSTALLATION

CerealBox models LV824-E and LV824-F have four connections that need to be made: power, 9 pin analog input, 25 pin digital input/output, and serial output. The LV824-G and LV824-H have an additional 9 pin connector for analog outputs. The CerealBox comes with a wall-plug power connector and a 25 ft. serial cable. You will need to manufacture your own cables with connectors for the analog and digital inputs/outputs.

Connect the wall plug power supply to a power outlet, and plug the coaxial jack into the case of the CerealBox. The CerealBox operates on 6.5 Volts to 9 Volts DC.

i *International Note. We provide US wall plug power supplies with the CerealBox. For those of you outside the US, you either need a plug adapter (if you already have 110 vAC), or purchase a 6.5 vDC wall plug power supply. Note the inner pin is positive.*

You can also provide power to the CerealBox via the cutout next to the analog input connector (see fig. 1). This requires an AMP connector (AMP 104257-1) with two pins (AMP 103358-6). The pin to the outside of the case is ground, and the inside pin carries 5 vDC which

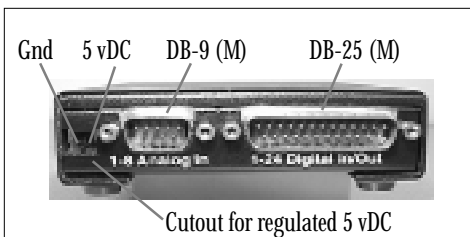


Figure 1 LV824 Inputs

MUST BE REGULATED. BG Systems can provide these connectors for a nominal charge.

Inputs to the CerealBox are made via the male DB-9 and DB-25 connectors at the end of the CerealBox as shown in figure 1. Analog inputs should be wired to the DB-9 connector with pins 1-8 for the input signals. Pin 9 must be grounded.

⚡ *Input voltages must be in the range of 0.0 to 5.0 volts. Higher voltage may damage the CerealBox!*

Digital inputs should be wired to pins 0-24 of the DB-25 connector. Pin 25 should be ground. Discrete inputs should be 0 volts off, 5 volts on. These inputs go to a TTL, so inputs over 3.0 volts will show as being on - we recommend using 5 volts.

For the LV824-F, the digital channels can provide 3.6 volt outputs as well. You should wire the inputs in groups of 8, and the outputs in groups of 8. (i.e., you could have pins 1-8 as inputs and 9-24 as outputs; or 1-8 as outputs, 9-16 as inputs, and 17-24 as outputs.) Whether the CerealBox produces outputs or reads inputs is configured in the software.

i *Before making output connections to equipment, you should test the output signal with a voltmeter to make sure that your physical wiring matches the software!*

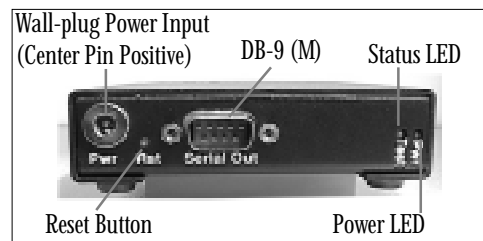


Figure 2 LV824-E and LV824-F

Serial output from the CerealBox to the host is from the male DB-9 connector next to the power input jack at the opposite end of the CerealBox as shown in figure 2. Pin 2 is for Rx, pin 3 is for Tx, and pins 5 and 7 carry ground. The connection on the host side will be either a DB-9 or a MiniDIN-8 connector. See section 4.1.1 for details of the types of serial cable.

For the LV824-G and LV824-H, there is an additional DB-9 (female) next to the serial output connector as shown in figure 3. Note that the connectors have different gender to prevent accidental connection of the wrong cable. For the LV824-G pins 1-3 are for the output signal, and pin 9 is ground. For the LV824-H pins 1-8 are for the output signal. As with discrete outputs, always check the values before connecting equipment !

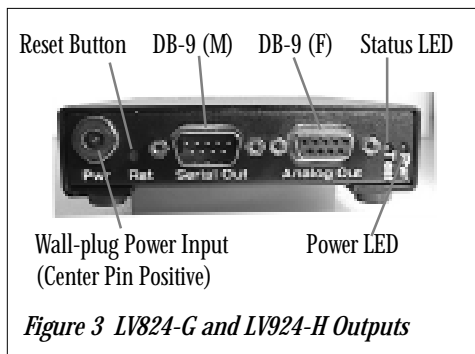


Figure 3 LV824-G and LV924-H Outputs

At the end of the CerealBox with the serial connector there are two LED's. The green LED indicates that there is power at the CerealBox. If this is not on, then either the power supply is bad, or there is a serious problem with the CerealBox.

The yellow LED blinks when power is applied. It blinks a "dot" once for a -E, twice for a -F, 3 times for a -G, and 4 times for an -H model. If you have encoder counters, the LED blinks in a "dash" once for a -4e, and twice for a -8e. The yellow LED will also blink

when a "T" is sent to the CerealBox (as in the *check_eprom.c* program), and it blinks faintly when the analog channels are being sampled. (If you are sampling at 30 Hz, the yellow LED glows.)

There is also a recessed reset button which requires a paper clip (or the end of a ballpoint pen) to activate. This button may be difficult to get to when power and the serial cable are connected - this is to prevent any inadvertent resets. This allows the CerealBox to be reset without turning power off and on.

i *The CerealBox should be placed as close to the equipment as possible, so that the cables carrying the input and output voltages are as short as possible. The serial cable between the CerealBox and the host is not affected by noise. But the cables between the CerealBox and the devices producing the signals are subject to noise and so should be as short as possible.*

i *Serial cables of up to 200 feet have been tested.*

The table below lists the pin-outs for the different connectors.

Function	Connector	Pin	Function
<i>Analog Input</i>			
	DB-9(M)	1-8 9	Analog Inputs Ground
<i>Digital Input/Output</i>			
	DB-25(M)	1-24 24	Digital Inputs / Outputs Ground
<i>Analog Output</i>			
	DB-9(F)	1-3 1-8	Analog outputs (-G) Analog outputs (-H)
<i>Serial</i>			
	DB-9 (M)	2 3 5 & 7	Rx Tx Ground
<i>5vDC Power Connector</i>			
	AMP	Left Right	Ground +5 vDC
<i>Encoder Counter Power</i>			
	AMP	Left Right	Ground +5 vDC

CHAPTER 3 SOFTWARE

This chapter describes the software interface to CerealBox. 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 highlight the differences in setup for the different versions of CerealBox.

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. Since the CerealBox can be used to collect data in a laboratory setting, and can also be connected to flight controls in a simulator, the range of applications and update rates is quite broad.

We provide source code for a number of example programs that interface to the CerealBox, which you should be able to modify to suit your needs.

i *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 3.1 reviews the basics of connecting the CerealBox to the host. Section 3.2 is a tutorial which studies an application that communicates with an LV824-E CerealBox. Sections 3.3 and 3.4 cover the differences between the basic CerealBox and the LV824-F, LV824-G and LV824-H versions. See section 4.4 of the **LV824 Software Manual** for information about using the encoder counters.

3.1 SOME BASICS

3.1.1 Connecting the CerealBox

Once you have physically connected the CerealBox to the host with an appropriate serial cable, and you have connected power to the CerealBox, you are ready to test the connection to see if you can communicate with the CerealBox. 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 physically connect the CerealBox. 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.

3.1.2 Theory of Operation

It is also useful to understand the basic sequence of events that are required to use the CerealBox. Since the CerealBox is an I/O device, we assume that the host needs to collect the inputs and send the outputs 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 CerealBox 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.

If you are using the output capabilities of a -F or -G CerealBox, the outputs are transmitted to the CerealBox at the same time as the request for data in step 1 above.

3.2 TUTORIAL E_TEST.C

Perhaps the easiest way to understand how to use the software is to study an example. In the *CerealBox/* directory, the source code in *e_test.c* is a fairly simple program that samples the CerealBox and prints the values to the screen. This section is based on chapter 3 of the Software Manual which should be consulted for more details.

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

```
% make e_t
```

to compile and link the executable *e_t*. Feel free to make these cryptic file names longer!

3.2.1 Setup

As you go through this example, look at the file *e_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
```

i *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    analog_out;
// Analog out selector - LV824-G
int    dig_out;
// Digital out selector, LV824-F & G
int    aout[3];
// Analog output data - LV824-G
int    dout[3];
// Digital output data - LV824-F & G
int    baud;
// Baud rate selected
```

i *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 CerealBox, the serial port needs to be opened by your program, and the appropriate configuration sent to the CerealBox. The function `setup_lv()` in the example performs this task.

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

The `analog_in` member of the `bgdata` structure sets any combination of the 8 available analog inputs. In the example above, channels 1, 2, and 4 have been requested by “or-ing” the predefined values `AIC1`, `AIC2`, and `AIC3`. Some other examples (if you understand the second you are in good shape):

```
bgdata.analog_in = AIC4 | AIC8;
// For channels 4 and 8
```

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

i *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 | DIC3;
```

In the example above, we have selected digital inputs 1-24. These are selected according to the following table:

<i><code>DIC1 0x10</code></i>	<i><code>Channels 1-8</code></i>
<i><code>DIC2 0x20</code></i>	<i><code>Channels 9-16</code></i>
<i><code>DIC3 0x40</code></i>	<i><code>Channels 17-24</code></i>

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

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

3.2.2 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/tty2",
            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 CerealBox is 19200**. This is important when you try to send initialization data to the CerealBox, 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.

i *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 CerealBox 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 7)
bgdata.Rev.alpha
// EPROM version e, f, g or h.
```

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 CerealBox:

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

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

i *Note that the CerealBox 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 CerealBox 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 CerealBox to 19200.*

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

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

(sample all 24 channels) and the CerealBox is an LV824-E, then the `init_lv()` call will fail

3.2.3 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.

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

3.2.4 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.

3.3 F_TEST.C FOR THE LV824-F

The next example to look at is for the LV824-F which allows you to set digital outputs. The 24 digital channels can be configured in groups of 8 as inputs or outputs in any combination.

3.3.1 Setup

In this example, we select none of the digital inputs:

```
bgdata.dig_in = 0x0;
```

and all of the digital outputs

```
bgdata.dig_out = 0x0;  
bgdata.dig_out = DOC1 | DOC2 | DOC3;
```

Of course you could select a combination of inputs and outputs, but you should not select the same group as inputs and outputs. (If you do, the *init_lv()* function will fail because of this conflict).

Next we set the value of all digital signals to zero:

```
bgdata.dout[0] = 0x0;  
bgdata.dout[1] = 0x0;  
bgdata.dout[2] = 0x0;
```

Setting the output bits operates as you might expect, you can set the output with an 8 bit hex number, or you can “mask” in values as you see fit. The file *pattern.c* includes some interesting ways to change the output bits.

3.3.2 Cycling

Since outputs have been selected, the *w_lv()* and *r_lv()* functions are no longer used. Instead:

```
send_outputs(&bgdata);  
// Also requests data back  
sigpause(SIGALRM);  
st = check_inputs(&bgdata);
```

The function *send_outputs()* uses the values in the *bgdata.dig_out[]* arrays to format a packet of data to send to the CerealBox. For example to set pins 4, 9, 10, 15, 23, and 24 to high:

```
bgdata.dout[0] = 0x08;  
// hex 8 sets pin4  
bgdata.dout[1] = 0x43;  
// hex 01 (p9), hex 02 (p10), hex40 p15  
bgdata.dout[2] = 0xc0;  
// hex 40 (p23) , hex 80 (p24)
```

Once the data is received by the CerealBox, the output lines are set to either 0 or 5 volts. After a suitable pause, the function *check_inputs()* is called. It is important to use this function, since it includes additional error checking in case the output to the CerealBox was not received correctly.

If an error is detected by the CerealBox, it requests a handshake from the *check_inputs()* function. This is to reduce the risk of problems when driving outputs which control mechanical devices. To illustrate this, there is a call to *w_lv()* after 51 cycles which includes an invalid message to the CerealBox - this triggers a handshake request.

```
w_lv(bgdata.sp_fd, "pE");
```

3.3.3 Testing

For testing purposes you could connect a switch to one pin, and an LED with a 1K resistor in series to another pin, and have the CerealBox turn the LED on and off. Of course the switch must be wired to an input group of 8, and the LED must be wired to an output.

You can also carry out loop-back testing, by connecting an output pin to an input pin. Of course the pins you choose have to be in different “banks of 8”.

3.4 G_TEST.C FOR THE LV824-G

This example is very similar to the *l_test.c* program, but it makes use of the analog output feature available with the LV824-G.

3.4.1 Setup

With an LV824-G, you can set analog outputs which drive voltages on the analog output connector. The first step is to determine which outputs are to be used:

```
bgdata.analog_out = AOC1 | AOC2 | AOC3;
```

In the example above we are going to use all three. Then you can set the initial values:

```
bgdata.a_out[0] = 0; // set pin 1
bgdata.a_out[1] = 0; // set pin 2
bgdata.a_out[2] = 0; // set pin 3
```

In the example above the voltages are all set to zero. The *init_lv()* routine will check that this setup is valid with the CerealBox.

3.4.2 Cycling


For the sake of producing an interesting output signal, we set the voltage to follow a sine wave:

```
si = sin( counter/100.0 );
bgdata.aout[0] = (int) ((1.0+si)*2048);
bgdata.aout[1] = 4095-bgdata.aout[0];
bgdata.aout[2] = 1023;
```

As with the LV824-F, we use *send_outputs()* and *check_inputs()* to write and read from the CerealBox:

```
send_outputs(&bgdata);
sigpause(SIGALRM);
st = check_inputs(&bgdata);
```

You can use a voltmeter connected to the analog output connector to verify that this example is running as expected.


 **You should use normal caution if using the outputs of a CerealBox to control dangerous equipment. Using a CerealBox to control equipment that may cause**

harm is not recommended by BG Systems, and if you choose to do so, you assume all risks.

The outputs voltages are set at the rate of 1 mVolt per bit. Since the resolution of the D/A chip is 12 bits, the range of voltages is:

```
bgdata.aout[0] = 0; // 0 volts
bgdata.aout[0] = 1000; // 1.0 volts
bgdata.aout[0] = 2500; // 2.5 volts
bgdata.aout[0] = 4095; // 4.095
volts (Max.)
```

Of course the length of the cable between the CerealBox and the device should be as short as possible to reduce the effects of noise.

 ***You should be able to draw up to 5 AMPS from each output.***

3.4.3 Loop-back Test

If you connect one of the analog output pins to one of the analog input pins, you can perform loop-back testing. You can “measure” the output voltage being sent with one of the input pins instead of a voltmeter. This is a test for all you software folks who don’t like hardware.

There is an example of this in the *examples/g_loop.c* file. Note that the input scale from -1.0 to 1.0 covers 5 volts, and the analog outputs have a range of 0.0 to 4.095 volts, so it the input *ain[0]* is scaled to print voltage on the screen. In this example, we drive each of the outputs with a different signal, and sample one input. By connecting analog out pin 1 to analog in pin 1 you can see the “voltage” change matched on the screen. Changing the connector to analog out pin 2 or three will show voltages that match the output signal.

3.4.4 h_test.c for LV824-H

This example is very similar to *g_test.c*, but sets the additional analog outputs available on the LV824-H.

3.4.5 Encoder Counters

See section 4.4 of the LV824 Software Manual.

4 TROUBLESHOOTING

We test every product prior to shipping, and have confidence that when you connect a CerealBox it will work. However, since we have two pieces of hardware (host and CerealBox), three pieces of software (host, CerealBox, and UNIX), and a cable, we realize that there will be occasions when you connect a CerealBox 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.

4.1 THE CEREALBOX IS NOT RESPONDING

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

- ⚡ *Is the green LED on? If not check your power supply.*
- ⚡ *Does the yellow LED blink when you turn on power? If not then the CerealBox is not working. Call BG Systems for help.*
- ⚡ *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 CerealBox. Check your cables and the host configuration.*
- ⚡ *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.*

The most common problem is that the host is configured with the requested serial port as a terminal running at 9600 baud. The CerealBox 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 CerealBox will stop responding.

4.1.1 Serial Cable

The most common 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 CerealBox to a new workstation for which you do

not have a BG serial cable proceed with caution. The table below lists the pin-outs for the various cables, and the on-line man pages also show these.

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

⚡ Remember that transmit (Tx) from the host computer connects to receive (Rx) on the CerealBox.

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 or move on to the next section.

5 TECHNICAL SPECIFICATIONS

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

A/D Resolution	12 bits
D/A Resolution	12 bits
Encoder Counter	24 bit resolution
TTL Voltage	3.6 vDC
Baud Rates Supported	2400, 4800, 9600, 19200, 38400, 57600, 115200
Power Supply Provided	9 vDC, 300 mAmp wall-plug power supply. 2.1 mm, center pin positive.
Power Draw (LV824-E)	240 mAmp
Serial Interface	RS-232 (Rx, Tx, Gnd)
Hardware Handshake	No



BG Systems, Inc. Palo Alto, California, USA

Tel: 650-858-2628

Fax: 650-858-2685

URL: www.bgsystems.com