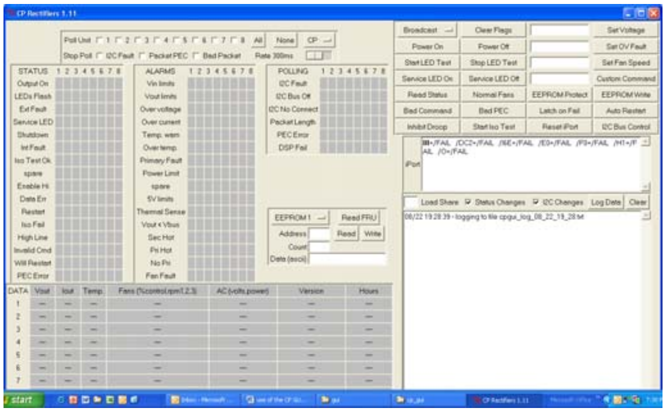
Application Note
April 2009

Compact Power Line (CPL)
Graphical User Interface (GUI) User Manual

Demonstrating the platform's i^{2}C based protocol

The interactive Graphical Interface is designed to demonstrate, and as a tool evaluate, the i^{2}C based communications capabilities of the Lineage CPL power system. The cpgui.exe file can run on any Windows computer. A version for Linux computers is also available on request. The Com1 port of the computer is utilized for exchanging information from the power module via an RS232 or USB to i^{2}C converter.

- Exercises the communications capability of the platform
- Capable of multi-tasking; status recording and instructing simultaneously.
- Automated data recording
- Automated error reporting and recording
- All recorded data is time stamped
- Configurable record keeping
- Automated display of bus traffic
- Records kept for up to 8 power supplies
- Automated RS232 port configuration
- USB port configuration



DATA	Vout	Iout	Temp.	Fans (%control,rpm1,2,3)	AC (volts,power)	Version	Hours
1	—	—	—	—	—	—	—
2	—	—	—	—	—	—	—
3	—	—	—	—	—	—	—
4	—	—	—	—	—	—	—
5	—	—	—	—	—	—	—
6	—	—	—	—	—	—	—
7	—	—	—	—	—	—	—

```
[illegible]+/FAIL  /DC2+/FAIL  /6E+/FAIL  /E0+/FAIL  /F0+/FAIL  /H1+/F
AIL  /O+/FAIL
```

```
08/22 19:28:39 - logging to file cpgui_log_08_22_19_28.txt
```

Table of Content

1 Introduction

Lineage Powers' Compact Power Line has an extensive communications instruction set. There are many unique commands and an equal number of read backs of status, alarms, inputs, outputs and temperature states of the power supply. There is also significant diagnostics capability built into the communications code. For example, the *isolation_test* command verifies that the output or'ing function is properly functioning in a working system. Information on individual fan speeds can be used to asses whether the fans are nearing their end of life. Firmware revision can attest to the capability of the product. A run timer keeps track of how long the power supply has been operational.

It would be a formidable task to ensure that all of these commands and read backs are properly functioning. The intent of the Graphical User Interface (GUI) is to provide the means for executing each command and enabling the read back of information from the power supply.

1.1 Enhancements

The CPL and this GUI are being updated and improved periodically. New capabilities are being retrofitted into existing products when time permits. Consequently, not all commands and read backs are implemented for all product. And, since new features are being retrofitted it is impossible to list the capabilities of each product because their capabilities are constantly improving. One thing is for sure, the latter products are better, support more capabilities, and are always backwards compatible. That is, existing commands will always be supported and will not change.

This issue of the GUI App note corresponds to the capabilities of rev 1.26 of the program.

1.2 Product demonstration

The Graphical User Interface makes the demonstration of capabilities task trivial and fun to perform. It is very user friendly and highly interactive. It displays the information in usable form and it automatically time-stamp records changes in events and outcomes for further analysis. The GUI also monitors the bus and ensures that transmission across the bus is not garbled or lost. If this should happen, the GUI will report these events in its display box. In summary, the GUI is a significant tool in demonstrating the product's interactive, diagnostics, and trouble reporting capabilities.

1.3 Product evaluation

Besides demonstrating the various command and response capabilities of the instruction set, the GUI has been the basic tool verifying whether a module is properly communicating. At Lineage, the GUI is extensively utilized during prove-in of the module firmware to ensure that stimulus, i.e. instructions, and their expected responses are properly handled by the software. Responses are evaluated by reading the state of the module. If a proper response is not received, it is back to the drawing board and changing of the software to ensure that the desired response is in fact provided.

On the surface this seems to be a trivial process. Quite the contrary, this is a tedious, repetitive and often exhausting process of trial and error until the desired outcome is achieved. And when it is finally working as expected, the integration test starts all over again. Because, not only do we need to prove in the response to a particular stimulus, we need to make sure that it did not affect anything else in the program.

Another very useful tool is really a Windows' based function, the ability to capture and copy the screen into another program, such as word or excel. Displayed data is thus saved into a systems integration report prepared by the development team for the product.

1.4 Utilization by OEM software developers

The GUI has been equally productive during the development of the end-use firmware by OEMs. OEMs utilize the GUI to verify whether the power module is properly communicating, especially during early prove-in of their software programs.

Often, developers initially have difficulty communicating with the power supply during their firmware development. On occasion, early-on during the development process, OEM written software includes minor interpretation errors that prevent proper communications. One of the very first questions of the OEM developer is what's wrong? Does the module work OK? Where does the problem lie? The GUI comes to the rescue by demonstrating communications capability and by tabulating the proper instruction set sent and received across the bus. With the GUI, the OEM software developer can verify whether the power supply properly communicates and is able to convince himself that the problem may be within the software he wrote. The display of the commands and the read back data, including the computed Packet-Error-Checking (PEC) byte, has been valuable in trouble shooting the new firmware.

The GUI is also used as a tool to verify how the power supply responds to external stimulus. By issuing the appropriate commands or by setting the proper state, such as interruption of input power to the module, the developer verifies the response of the power supply. The software developer uses this information to interpret the condition and execute the desired action.

1.5 Record keeping

Execution of the GUI program automatically creates a data file in the same directory where the GUI executable is located within the computers' file structure. Each data file is uniquely identified. The data file automatically closes whenever the GUI is closed on the machine.

The GUI is able to exercise all of the I2C commands understood by the power supply and it monitors all of its states. It can be placed into an automated mode of read-back, called polling, and into a state where each module is sequentially evaluated. This evaluation compares the state of the module to its previously read state. If the two states do not equal, the GUI can be set to automatically record and time stamp the event when this state change has occurred. The time stamp includes the date, time, and the read back status and alarm registers that are active HI.

Other events such as the analog readings that are displayed on the screen can also be recorded and saved for later investigation or reporting. For example, current share between modules can be recorded and later extracted for evaluation or reporting.

2 Protocol

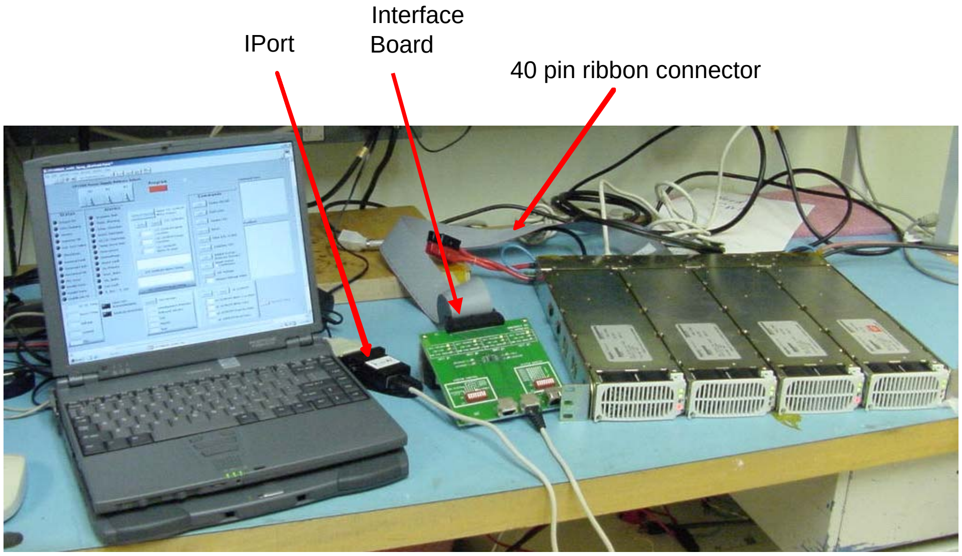
The GUI is designed to exercise the instruction set documented in the **i^{2}C Protocol for the CPL Product Line.** This document may be obtained for reference from either Lineage Sales or the FAE team.

The Protocol is evolutionary and new features are folded into the product when time and schedule permit such improvements.. Because of the evolutionary nature of the command set not all commands work on all modules. Typically, if a certain command is not supported the module would raise the **invalid_command** bit of the STATUS registers.

3 Required components

A basic setup is composed of the following:

- A computer with either an RS232 interface or a USB port with a USB to RS232 cable set,
- A commercially available Translator from Micro Computer Control (mcc-us.com) either the MIIC-202 RS232-i2c host adapter or the MIIC-204 USB – i2c host adapter.
- an Interface board that mates to the i^{2}C cable of the IPort on one end and to the Lineage Power shelf on the other end, a 40 position ribbon cable set,
- a Lineage Power shelf or equivalent,
- and at least one power supply



Setup of the GUI and the CPL power system

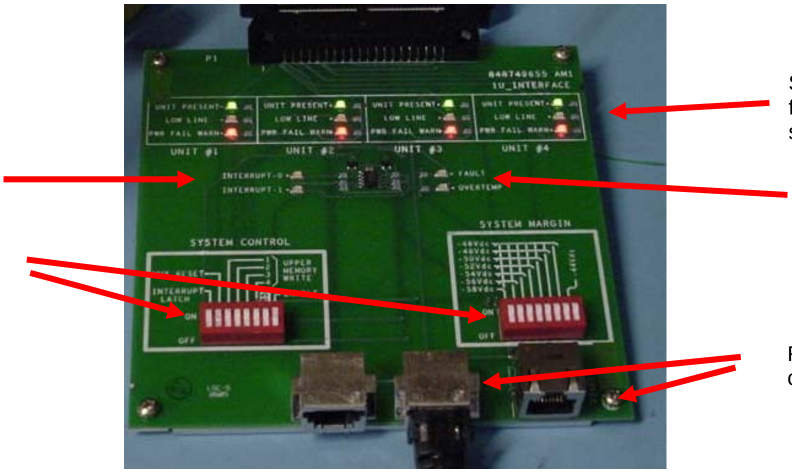
3.1 The Host Adapter

The *Host Adapter* accepts ASCII instructions from the computer's RS232 port and converts the instruction set to i2c standard signals. The chosen adapter is **IPort** model # **MIIC-202** manufactured by Micro Computer Control Corporation. (www.mcc-us.com). Other IPort variations are not recommended.

The IPort was designed to communicate to a single device. See the appendix for recommendations to change the pull-up resistors within the IPort if multiple devices are going to be used on the bus. The IPort has a sliding switch on one side. Make sure that the switch is set to the ON position. This connects the pull-up resistors of the IPort into the data and clock lines.

3.2 The CPL Interface Board

Designed by Lineage as an interface between the Lineage shelf and the user, LEDs read signaling states and micro-switches write commands to exercise the analog capability of the product. RJ type connectors interface with either the RS485 based protocol or two independent i^{2}C based protocols.



Interrupt/Alert Signals

Control switches

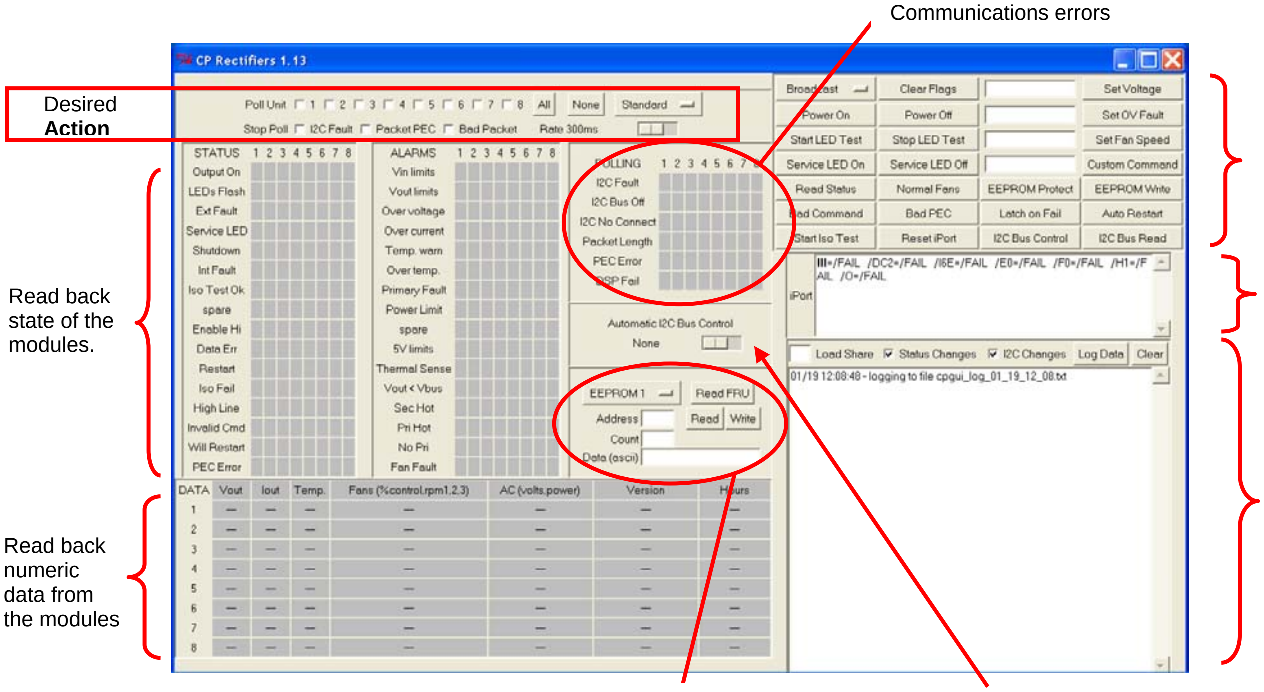
Signals displayed from the four power supplies

Combined Fault and Over-temperature signals

RS485 and i^{2}C communications

4 GUI Display areas

The GUI display is partitioned into eight different areas. Each of these areas is explained below in further detail.



Communications errors

Desired **Action**

Instructions

Read back state of the modules.

Transmitted data

Display of state changes and selection of saved information

Read back numeric data from the modules

EEPROM instructions

Repeat interval of auto bus control comand

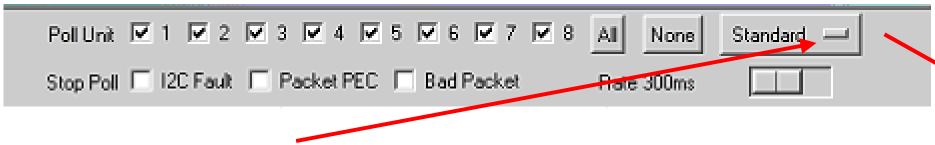
```
III=/FAIL  /DC2=/FAIL  /I6E=/FAIL  /E0=/FAIL  /F0=/FAIL  /H1=/F
AIL  /O=/FAIL
```

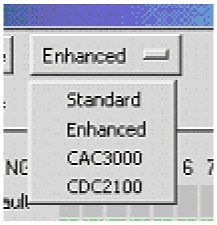
```
01/19 12:08:48 - logging to file cpgui_log_01_19_12_08.txt
```

DATA	Vout	Iout	Temp.	Fans (%control,rpm1,2,3)	AC (volts,power)	Version	Hours
1	—	—	—	—	—	—	—
2	—	—	—	—	—	—	—
3	—	—	—	—	—	—	—
4	—	—	—	—	—	—	—
5	—	—	—	—	—	—	—
6	—	—	—	—	—	—	—
7	—	—	—	—	—	—	—
8	—	—	—	—	—	—	—

4.1 Desired Action

This section determines which modules are going to be continuously polled and what revision of the product is being polled.


Poll Unit ☑ 1 ☑ 2 ☑ 3 ☑ 4 ☑ 5 ☑ 6 ☑ 7 ☑ 8
All
None
Standard
Stop Poll □ I2C Fault □ Packet PEC □ Bad Packet
Rate 300ms



Clicking on the small box inside the Polling Type rectangle brings up a dropdown menu that can select one of four features; Standard – displays the read back that is consistent throughout the platform, Extended – displays in addition to the standard read back the latest features of the product such as fan speed control, firmware revisions, input voltage and power readings (CP2725 only), and run timer , CAC3000 – displays the extended read back option of this code which has a different constant for current conversion, and CDC2100 – displays the extended read back for this code with a different set of status and alarm register definitions.

4.1.1 Unit selection

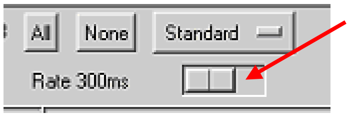
Clicking inside the white square area to the left of unit # starts polling of the module in that slot. When polled a check mark appears in the box. Alternatively, clicking the ALL button automatically selects the polling of all 8 modules. Conversely, clicking on the NONE button stops polling of all modules and additionally clears the displayed information. Unit #1 is the first module of shelf-0 [1000001x], and unit #5 is the first module of shelf-1 [1000101x]. (See further information on unit and shelf addressing in the protocol spec.)

4.1.2 Stop polling when an error is detected

Polling can be stopped for up to three different error types by clicking inside the appropriate square area next to the type listed; i2c fault – typically a transmission problem, Packet PEC – failure of the comparison between the PEC calculated by the GUI and the PEC imbedded in the message from the module, Bad Packet – is an error in the number of bytes received or some other transmission error. When selected, a check mark appears inside the box.

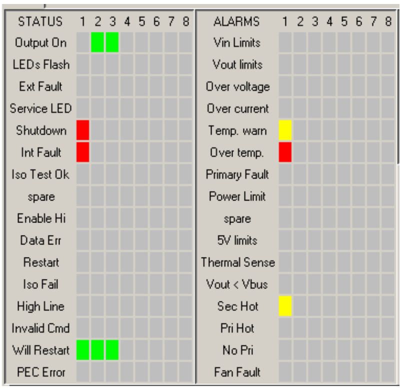
4.1.3 Polling rate

The read back rate can be changed on the fly from 200ms to 5 seconds. Click and drag the square area left or right or click in the space between the square and the end of the guide. The read back rate is always displayed for the user. Reading too fast causes the display of intermediate, transitional, events without providing any further meaningful data. Read backs should not be attempted any faster than 2 seconds and at turn ON a delay of about 30 seconds is desired to complete the initialization process.



4.2 Read Back – Binary Registers

The STATUS of all 8 modules is continuously displayed to the user. The displayed color assesses event severity. Green – normal, Yellow – warnings, Red – fault requiring immediate attention.



STATUS	1	2	3	4	5	6	7	8
Output On		🟩	🟩					
LEDs Flash								
Ext Fault								
Service LED								
Shutdown	🟥							
Int Fault	🟥							
Iso Test Ok								
spare								
Enable Hi								
Data Err								
Restart								
Iso Fail								
High Line								
Invalid Cmd								
Will Restart	🟩	🟩	🟩					
PEC Error								

ALARMS	1	2	3	4	5	6	7	8
Vin Limits								
Vout limits								
Over voltage								
Over current								
Temp. warn	🟨							
Over temp.	🟥							
Primary Fault								
Power Limit								
spare								
5V limits								
Thermal Sense								
Vout < Vbus								
Sec Hot	🟨							
Pri Hot								
No Pri								
Fan Fault								

In the display to the left, module #1 has been shutdown because of an over-temperature event on the secondary side. The module assesses that this is an internal fault which is the likely cause but not necessarily correct. The thermal condition could be caused by either an excessive thermal ambient or airflow blockage. These events are beyond the assessment capability of the module electronics.

Below is an excerpt from the i2c protocol defining the 32 unique bits of the STATUS and ALARM registers.

Register	Bit	Title	Description
Status_2 (ON=1=true)	7	PEC Error	The computed PEC does not match the transmitted PEC. The instruction has not been executed.
	6	Will Restart	The power supply will restart after a shutdown
	5	Invalid Instruction	The instruction is not supported. An ALERT# will be issued. Clear_Flags resets this register.
	4	High Power Capacity	High line power capacity of power supply
	3	Isolation test failed	This flag requires replacement of the power supply
	2	Restarted ok	Informs *HOST* that a successful *RESTART* occurred clearing the status and alarm registers
	1	Data out of range	Flag appears until the data value is within range. ALERT# clears after a READBACK. A clear_flags command does not reset this register.
	0	Enable pin HI	Power supply instructed to turn OFF via hardware

Register	Bit	Title	Description
Status_1 (ON=1=true)	7	spare	
	6	Isolation test OK	Indicator that the Isolation_Test requested has been completed successfully.
	5	Internal fault	The power supply is faulty
	4	Shutdown occurred	The power supply is shut down
	3	Service LED ON	Used to identify a particular power supply
	2	External fault	The recorded fault is external, the power supply is OK
	1	LEDs flashing	Tests the LED display of the power supply
	0	Output ON (power good)	

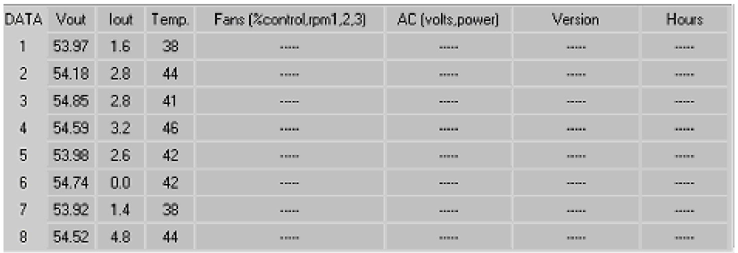
Register	Bit	Title	Description
Alarm_2 (ON=1=true)	7	Fan Fault	A fan failure causes a power supply latched shutdown
	6	No primary detected	Internal fault of the primary section of the power supply
	5	Primary over temperature	Fault detection of either the boost or primary of DC/DC
	4	DC/DC over temperature	Secondary heat sink temperature of the DC/DC stage
	3	Output voltage lower than bus	Internal regulation failure
	2	Thermal sensor failed	Internal failure of a temperature sensing circuit
	1	5V out_of_limits	Either OVP or OCP of the 5V circuit has tripped
	0	Spare	

Register	Bit	Title	Description
Alarm_1 (ON=1=true)	7	Unit in power limit	An overload condition that results in constant power
	6	Primary fault	Indicates either primary failure or INPUT not present. Used in conjunction with bit-0 and Status_1 bits 2 and 5 to assess the fault.
	5	Over temp. shutdown	One of the over_temperature sensors tripped the supply
	4	Over temp warning	Temperature is too high, close to shutdown
	3	**In over current**	Shutdown is triggered by low output voltage < 39Vdc.
	2	Over voltage shutdown	
	1	Vout out_of_limits	Indication the output is not within design limits. This condition may or may not cause an output shutdown.
	0	Vin out_of_limits	The input voltage is outside design limits

4.3 Read Back – Analog Data

4.3.1 Standard display

The *standard* display records output voltage, output current, and internal temperature of the module. In addition the four state registers of status and alarm, and the state of communications are also displayed.

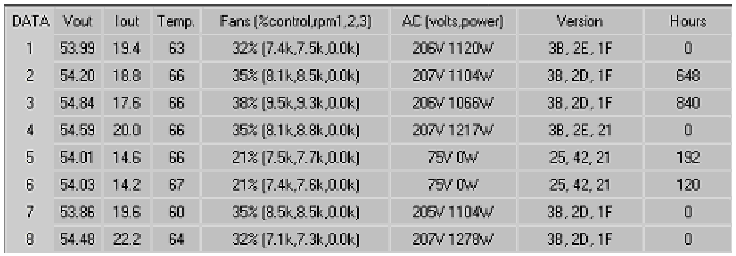


DATA	Vout	Iout	Temp.	Fans (%control,rpm1,2,3)	AC (volts,power)	Version	Hours
1	53.97	1.6	38	-----	-----	-----	-----
2	54.18	2.8	44	-----	-----	-----	-----
3	54.85	2.8	41	-----	-----	-----	-----
4	54.59	3.2	46	-----	-----	-----	-----
5	53.98	2.6	42	-----	-----	-----	-----
6	54.74	0.0	42	-----	-----	-----	-----
7	53.92	1.4	38	-----	-----	-----	-----
8	54.52	4.8	44	-----	-----	-----	-----

The data to the left shows one of the drawbacks of fast record keeping. The data seems to indicate that current share is not working very well. In fact, the current share loop is purposely slow and it may take up to a minute for the modules to stabilize and current share properly. Continuous polling would show the convergence of current share among the modules.

4.3.2 Extended (or code specific) display

These are newer features that have been incorporated only on the latest products and are retrofitted slowly into earlier models such as the CP1800 and CP2000.



DATA	Vout	Iout	Temp.	Fans (%control,rpm1,2,3)	AC (volts,power)	Version	Hours
1	53.99	19.4	63	32% (7.4k,7.5k,0.0k)	206V 1120W	3B, 2E, 1F	0
2	54.20	18.8	66	35% (8.1k,8.5k,0.0k)	207V 1104W	3B, 2D, 1F	648
3	54.84	17.6	66	38% (9.5k,9.3k,0.0k)	206V 1066W	3B, 2D, 1F	840
4	54.59	20.0	66	35% (8.1k,8.8k,0.0k)	207V 1217W	3B, 2E, 21	0
5	54.01	14.6	66	21% (7.5k,7.7k,0.0k)	75V 0W	25, 42, 21	192
6	54.03	14.2	67	21% (7.4k,7.6k,0.0k)	75V 0W	25, 42, 21	120
7	53.86	19.6	60	35% (8.5k,8.5k,0.0k)	205V 1104W	3B, 2D, 1F	0
8	54.48	22.2	64	32% (7.1k,7.3k,0.0k)	207V 1278W	3B, 2D, 1F	0

The data here shows a number of features. The precision of current share is better demonstrated by waiting until the loop had the time to stabilize. The readings also show the proportional current share capability of the CPL platform. Units 5 and 6 are the latest CP2000AC54 modules while the rest are CP2725 modules. The currents sourced by the CP2000 modules are lower because it is proportioned based on its maximum capability. That is, all units deliver an equal percentage proportion based on their full ratings. Also note that the input voltage and power does not display meaningful information for the CP2000 modules. Firmware revision numbers are also different since we used whatever modules we could get our hands on for this demo.

The *extended* display adds the following new features;

- Commanded fan speed and the speed of individual fans – The commanded speed is expressed in percent of full speed. The fans cannot spin slower than about 30%. Commands below that percentage will yield the 30% minimum speed. The speed of up to three fans is reported. If only two fans exist in a power supply, then the third fan speed read back is 0.0k.

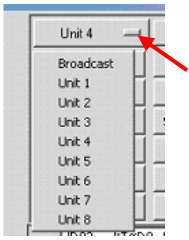
- Input source voltage and input power consumed by the module – Only the CP2725AC54 rectifier provides this new feature.

- Software revision of the primary µController, the DSP and the i^{2}C µController in that order. The information is in hex code. It is read as 1.6 when 16 is displayed. 1F is equivalent to rev 1.15.

- Hours of operation – A run timer keeps track of how long the power supply has been operational. The DSP saves in memory only one count for every 24 hours of operation. This truncated count is required in order to stay within the maximum write cycle of the device and provide the desired minimum 10 year count. For example, if the power supply is ON for 23.5 hours and then its input is turned OFF (no external bias present), then at turn ON the count starts all over at 0. If the power supply is ON for 33 hours and then it is turned OFF, then at turn ON the count starts at 24.

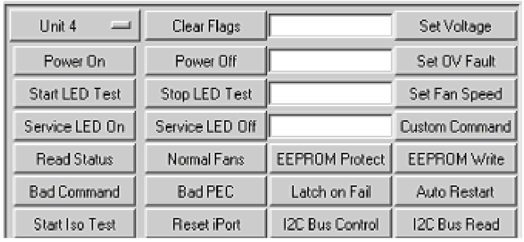
As we stated earlier, the extended read back is gradually added on to the platform. If not supported then the read back is 00, with the exception of input voltage which returns 75 (the equation for input voltage takes in a variable and adds 75 for accuracy improvements, so the lowest number is the +75) . In the above example fan and input read backs are not supported. In addition the revision of the primary µController is not supported. On some earlier models, non-supported commands of the extended read back will flag the *invalid_command* bit of the STATUS registers.

4.4 The Instruction Set



This section of the GUI contains all the instructions that a module could respond to. Note that there is a little rectangle to the right of the Broadcast box. The GUI defaults to the Broadcast address which sends out the instruction to all modules on the bus. Clicking on the rectangle brings up a drop down menu. Clicking on the unit # of this drop down selects an individual unit to be instructed.

There are a number of commands that must be instructed to individual modules and these are: Read Status, and Start Isolation Test.



Each command is summarized below; (Note: location of the buttons may be different in previous revisions of the GUI)

4.4.1 Power ON / Power Off

This is self explanatory. The main output of the power supply is commanded to turn ON or OFF via these commands.

4.4.2 LED Test

Turns ON and OFF all front LEDs of the module. The ON and OFF times may vary

4.4.3 Service LED

Turns ON and OFF the service LED on modules that have this feature. The ON state is blinking.

4.4.4 Set voltage, OV

In these boxes the setpoint and the OV shutdown point of the module can be changed.

4.4.5 Inhibit Droop

Instructs modules supporting this feature to maintain a constant output voltage instead of a voltage level that is proportional to delivered output current.

4.4.6 Set fan speed

On modules supporting this feature the fan speed can be increased above that commanded by the DSP controlling the module.

4.4.7 Normal fans

Instructs the module to change the fan speed back to the level required for normal module operation. Reduction in fan speed is a very slow process taking up to a few minutes to complete.

4.4.8 Custom command

This feature is used by the Lineage development team.

4.4.9 Read Status

Executes a single read instead of the polling option

4.4.10 Bad PEC / Bad Command

Sends incorrect instructions to the module. A read back should verify that the incorrect instruction has been identified.

4.4.11 Clear Flags

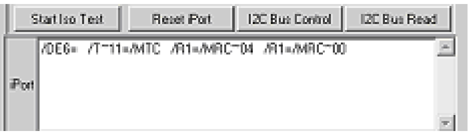
Informative STATUS bits are reset by the controller via this command. Informative bits include Data out of range, Invalid data, and PEC error among others.

4.4.12 I2c Bus Control

Instructs the module to take over bus control to the connected line. This is the command that shifts communications between i2c-0 and i2c-1.

4.4.13 I2c Bus Read

This command reads back the state of the PCA9541 multiplexer in two steps, first it reads the status register and then the interrupt register. There are two independent reasons for commanding an I2c Bus Read. The first is to read back the state of the multiplexer. The second is the clear the *interrupt/Alert#* signal from the module if the PCA9541 set the interrupt in the first place. (A readback of the interrupt register of the communicated channel automatically clears the interrupt signal of that channel. If the interrupt does not clear than the signal is set by the i2c micro controller and not the multiplexer).



```
/DE6=  /T~11=/MTC  /R1=/MRC~04  /R1=/MRC~00
```

The Bus Read instruction is always the command 0x11, which reads the status register first and then increments to the interrupt register. MTC – is the master transmit complete indicating that the instruction was properly sent. The two read back are the data from the status followed by the interrupt register. This information is presently not recorded in the data record.

Information from the status register needs to look at only the least 5 bits. Note that the information is in hex so a hex to binary conversion needs to take place to interpret the information in the register.

Bits	Meaning
4	1 – indicates that bus initialization was completed 0 – indicates that bus initialization was not completed
2 - 3	1,0 or 0, 1 indicates that the bus is ON 1,1 or 0,0 indicates that the bus is OFF
0 - 1	1,1 or 0,0 indicates that this channel is in control 1,0 or 0,1 indicates that this channel is not in control

In the data above the status readback is 0x04 which is converted to 00100 (5 least significant bits) and indicates that the Bus is ON and that this channel is in control.

Data translation of the interrupt register can be obtained from the PCA9541 multiplexer data sheet.

4.4.14 Reset IPort

Used to re-establish communications between the computer and the IPort if a lock-up occurs.

4.4.15 Latch on fail

Commands the module to stay latched after a shutdown. A restart from latchoff typically requires powering OFF and then ON either via software or hardware.

4.4.16 Auto restart

Commands the module to restart automatically as programmed

4.4.17 Isolation test

Commands the module to execute a test of its output or’ing function and the input or’ing function as well on modules such as the CDC2100 that support input or’ing.

4.4.18 EEPROM write / protect

Instructs the module to either permit or deny a write instruction into the upper ¼ of memory.

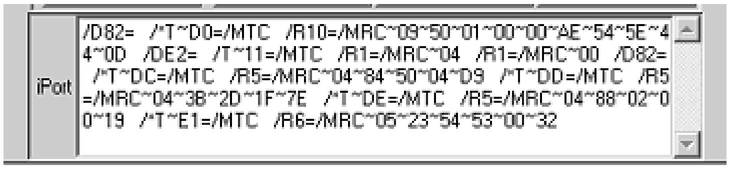
One important feature of the GUI is that POLLING continues when an instruction is executed. The instruction is executed instead of a single read back and then the read back resumes where it left off.

4.5 Bus Activity

Commands and Instructions to and from the module are recorded in the Bus Activity area of the GUI. This is quite useful when debugging transmission or instruction related problems.

Every character in the transmission below means something. The data fields are documented in the i2c protocol document.

In the recorded activity displayed below, device 82 [Unit #2 – 10000010] is first requested to read status [D0], 10 bytes of data are received, the first byte tells the host that 9 data bytes follow. When this instruction is completed the GUI addresses the internal multiplexer of the module device E2 and command 11 is executed. (reads the state of the multiplexer and increments to read the state of the interrupt register, see for further information the data sheet for the NXP PCA9541 multiplexer). When this instruction is complete device 82 is again addressed and the remaining read backs (fan control, Software revision, Input, and Run time) are requested one at a time. Each data file starts with a byte stating how many bytes of data follow. The last data byte of each data packet is always the computed PEC.



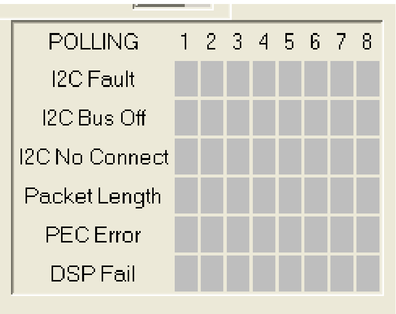
```
/D82=  /*T~D0=/MTC  /R10=/MRC~09~50~01~00~00~AE~54~5E~4
4~0D  /DE2=  /T~11=/MTC  /R1=/MRC~04  /R1=/MRC~00  /D82=
 /*T~DC=/MTC  /R5=/MRC~04~84~50~04~D9  /*T~DD=/MTC  /R5
=/MRC~04~3B~2D~1F~7E  /*T~DE=/MTC  /R5=/MRC~04~88~02~0
0~19  /*T~E1=/MTC  /R6=/MRC~05~23~54~53~00~32
```

MTC – Master Transmit Complete
MRC – Master Receive Complete

4.6 State of Communications

4.6.1 Fault annunciation

Previous revisions of the GUI recorded only communication faults across the bus. The messages reported the following:



POLLING	1	2	3	4	5	6	7	8
I2C Fault								
I2C Bus Off								
I2C No Connect								
Packet Length								
PEC Error								
DSP Fail								

I2C Fault – Communications failed with either the I2C µController or EEPROM. Typically an address is no-ack'ed.

I2C Bus OFF – Communications are not established on this bus.

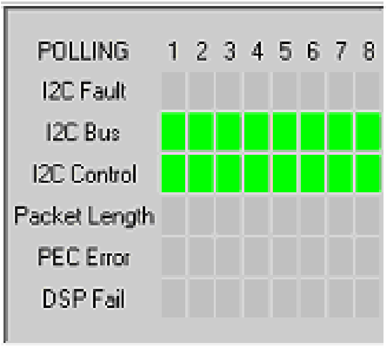
I2C No Connect – The module multiplexer is not connected to this I2C line.

Packet Length – Length of the data packet is incorrect.

PEC error – The calculated PEC does not agree with the transmitted PEC.

DSP Fail – If internal communications between the DSP and the I2C µController fail then the µController transmits FF's in the STATUS and ALARM registers and the PEC calculation. This bit is set when such a transmission of FF's is recorded.

4.6.2 Indication of having control

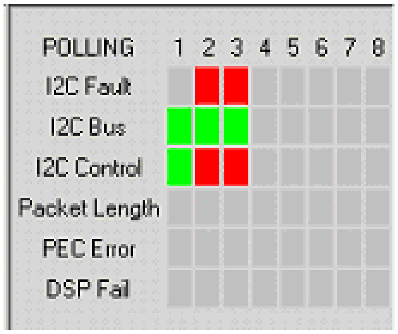


POLLING	1	2	3	4	5	6	7	8
I2C Fault								
I2C Bus	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
I2C Control	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
Packet Length								
PEC Error								
DSP Fail								

Current revision of the GUI records all communications states, not only error messages. This way we have information on when communications are also properly established rather than guess whether messages are properly being sent across the bus.

When communications are properly estabished to each power supply the I2C Bus and I2C Control indicators are both green. These indicators are read from the PCA9541 multiplexer that is connected to the two independent i2c lines of the power supply and switches between i2c-0 and i2c-1.

4.6.3 Indication of no-control

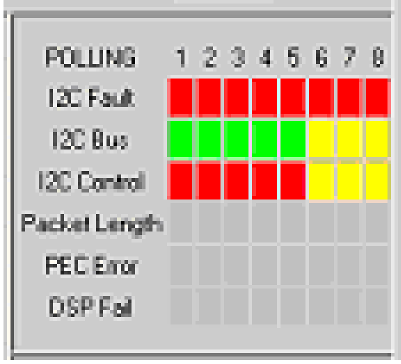


POLLING	1	2	3	4	5	6	7	8
I2C Fault		🟥	🟥					
I2C Bus	🟩	🟩	🟩					
I2C Control	🟩	🟥	🟥					
Packet Length								
PEC Error								
DSP Fail								

The alternate of having control of the bus is not having control. The latest GUI properly reports this event as well. This is not an alarm condition but a report of the state of the bus normally when either two controllers (GUIs) are connected or when a controller is connected to an i2c line that is not connected to the power supply.

The indicators on the left tell us that the GUI cannot communicate across the I2C line for modules #2 and #3. Inquiry of the state of the PCA9541 multiplexer interprets that the multiplexer bus is ON, but it is not ON one the present I2C line. The normal procedure would be to issue an I2C_Bus_Control command to take over control of the I2C bus on this connection. Note that module #1 is connected and has control of this I2C bus. Each module has its own independent multiplexer and so each module can be instructed to be switched to either I2C line.

4.6.4 Indication of no response



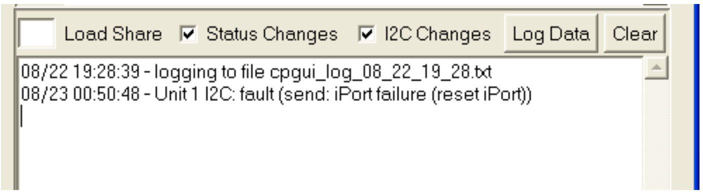
POLLING	1	2	3	4	5	6	7	8
I2C Fault	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥
I2C Bus	🟩	🟩	🟩	🟩	🟩	🟨	🟨	🟨
I2C Control	🟥	🟥	🟥	🟥	🟥	🟨	🟨	🟨
Packet Length								
PEC Error								
DSP Fail								

This condition is likely to occur if the GUI is not successful in reading back the communication status of the PCA9541 multiplexer. For example, The PCA9541 no-ack's its address.

The interpretation here is that communicating with the multiplexer has not been established. The state is *yellow* because so many conditions can cause this type of response that it cannot be determined that an error has occurred. For example, a non-powered shelf or an empty slot could cause this condition.

In the example in the left it is likely that slots 6 -8 are empty. Slots 1 – 5 indicate no-control of this bus.

4.7 Generation of the data file



□ Load Share ☑ Status Changes ☑ I2C Changes

```
08/22 19:28:39 - logging to file cpgui_log_08_22_19_28.txt
08/23 00:50:48 - Unit 1 I2C: fault (send: iPort failure (reset iPort))
```

One of the most useful features of the GUI is the automatically generated text file that registers and time stamps all changes in activity during the time the GUI is exercised. Properly utilized, this file maintains a historical record of the particular exercise. The GUI can be left unattended and the data file can be reviewed later for any changes in activity.

Three different types of events can be time stamp recorded. Note the time stamp below, it includes the date, hours, minutes, and seconds.

The three event types are:

- Status changes – Whenever any of the 32 bits of the Four STATUS and ALARM registers change states from the previous read, the event is recorded.

- I2C changes – Commands, errors and information about the state of the bus are recorded

- Load share – if a number is entered in this box, the GUI will verify whether the current reading of a module is higher or lower by the number entered the averaged current readings of the modules being tested. For example; If a number 5 is entered, the GUI will verify whether an individual reading is higher or is lower by 5 amperes the average of all modules. The GUI will time stamp and record the deviation in the data file.

4.7.1 Log Data

Clicking on this icon results in the time stamped recording of the analog data displayed in the GUI.

The time stamped data below recorded the extended data of two CP2725 rectifiers

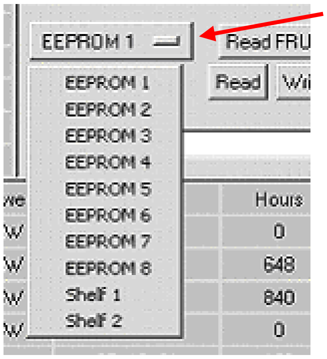
04/02 15:18:06 - Unit 1 Vout=54.02 Iout=1.6 Temp.=37 Fans (%control,rpm1,2,3)=84% (18.5k,19.0k,0.0k) AC (volts,power)=209V 181W Version=3B, 2E, 1F Hours=1

04/02 15:18:06 - Unit 2 Vout=54.16 Iout=3.2 Temp.=39 Fans (%control,rpm1,2,3)=84% (18.6k,18.9k,0.0k) AC (volts,power)=208V 300W Version=3B, 2D, 1F Hours=649

4.7.2 Clear

Clicking on this icon clears the recorded data area. This action does not clear the contents of the text file.

4.8 EEPROM write and read

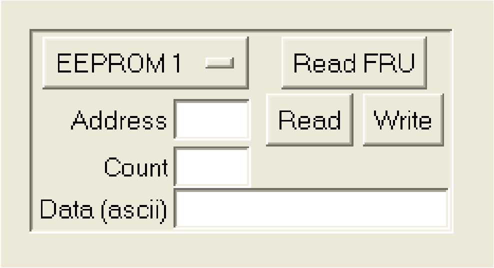


Click on the drag down rectangle and select either the EEPROM of the module or the EEPROM of one of two shelves. The two shelves are custom shelves configured as Field Replacable Unit (FRU) at the customer's request with 2kbit memory EEPROMs without a protected area.

The **Read FRU** button provides an easy way of accessing the FRU-ID information in the EEPROM section of the module with a single click of a button.

The module EEPROMs are 64kbit memories having a capacity of 8kbyte words. The upper ¼ of memory (6 – 8 kbytes) are in the write protected area. The Address is a hex word. Use the computer's calculator to determine the hex equivalent of the desired address location. (Example: The FRU-ID information starts at decimal memory location 6144, its hex equivalent is location 1800.)

All EEPROM commands are recorded automatically in the data file.



To **write** information into the EEPROM the starting memory address has to be entered in hex format. (You can easily convert from decimal to hex and reverse using the scientific calculator on your computer). The DATA entered has to be ASCII characters. If no write permission is given data in the protected area cannot be overwritten.

To **read** information from the EEPROM, enter the starting memory address and the number of bytes being read back.

4.8.1 Quick reading of the FRU-ID section

The example below shows a reading of the FRU-ID of a standard Lineage module. First EEPROM 3 was selected (this is the EEPROM section of unit #3) and then the Read_FRU button was clicked. If the execution is successful the following information will display in the data section and recorded automatically into the data file;

```
04/02 15:28:26 - Read EEPROM 3 FRU Information...
04/02 15:28:27 -      pid=CP2725AC54.. pn=C109142873 hw=03     sw=01
04/02 15:28:27 -      sn=08LD19500957.. manu=Dongguan, Guangdong, China.............
04/02 15:28:27 -      read_volts_mbr=400/0/0 read_amps_mbr=5/0/0 read_temp_mbr=1/0/0
04/02 15:28:27 -      set_volts_mbr=400/0/0
```

4.8.2 Manual access and read back

Instead of reading back the canned program embedded in the Read_FRU subroutine, the information can be accessed manually. In this example, EEPROM 3 still has to be selected, but instead of clicking on the Read_FRU button, enter in the address box 1900, enter in the count box 32 for 32 bytes of information and click on the READ button.

```
04/02 15:22:51 - Write EEPROM 1 @ 1900: This is Gabe talking
04/02 15:22:54 - Read EEPROM 1 @ 1900: 54 68 69 73 20 69 73 20 47 61 62 65 20 74 61 6C 6B 69 6E 67 1 90 0 0 0 0 1 0 4B 0
54 0  'This is Gabe talking........K.T.'
```

4.8.3 Writing and reading

Writing into the unprotected area can be done at any time. Writing into the protected area has to be preceded by instructing the module to enable the writing capability by clicking on the **EEPROM Write** button of the command section. In this example, EEPROM 3 still has to be selected. To write information into the EEPROM section, first identify the EEPROM address where the content writing should start, in our case address 1500, and then enter in the DATA section 'This is Gabe talking' and click on the **Write** button. (note that for writing you do not need to enter a byte count).

To read back the information keep the starting address box in the box ar 1500, enter in the count box 32 for 32 bytes of information to be read back, and click on the READ button. The read back first records all the memory information in hex format and then the program translates the information into ASCII code. (note that the information was significantly less than 32 bytes and therefore the dots followed indicating that the information in these locations is not in ASCII format.

```
04/02 15:21:09 - Send to Unit 1 (80): EEPROM Enable (D6 9A )
04/02 15:21:43 - Write EEPROM 1 @ 1500: This is Gabe talking
04/02 15:21:51 - Read EEPROM 1 @ 1500: 54 68 69 73 20 69 73 20 47 61 62 65 20 74 61 6C 6B 69 6E 67 FF FF FF FF FF FF FF
FF FF FF FF FF  'This is Gabe talking...........'
```

5 The IPort interface

The IPort translates into proper i^{2}C commands bidirectional instructions and data that originate from either the RS232 (IPort/AI) or USB I(USB_IPort) communications ports of the computer. These are the selectable hardware ports that the GUI is communicating to.

The driver for the IPort must be loaded onto the computer before any communications can take place, the proper communications port has to be identified and the GUI needs to be directed to communicate to the identified communications port. The instructions below go through a step by step process on how to set up a computer if the communications port is anything but COM1. If COM1 is selected then these steps are not needed.

5.1 Loading a driver

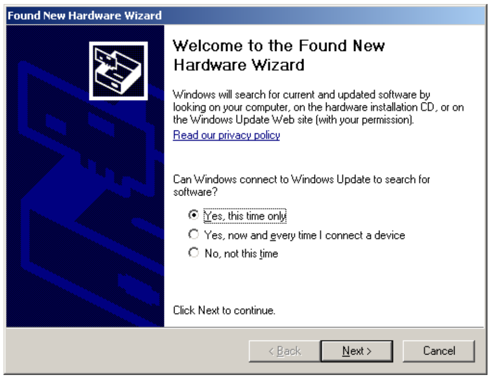
The driver can be loaded from the disk supplied by MCC or by downloading the driver from their website at //mcc-us.com.

To download the USB based driver click on **Download (Upgrades/Updates)** wait until the screen changes and then click on **iPort/USB(#MIIC-204)** wait until the screen changes then click on **iPort Message Center V5.4.2** wait until the screen changes and then navigate down to the bottom of the screen and click on **Download iMsgCtr542.Zip** wait for the screen to change and then download the zip file to **C:\Program Files\MCC**. If this directory does not exist, then create it.

Open the Zip file, click on **Actions** and then on **select all** and **extract** the files into the same directory, **C:\Program Files\MCC.** You should have created two directories under MCC, **iPort USB** and **IPortDLLUSB**,

5.2 Installing the driver

Plug in the USB connector of the IPort_USB into any available slot. The computer should recognize that an unknown hardware has been inserted and should come up with the screen below; click on *Yes, this time only* and click on **< NEXT >**



Welcome to the Found New Hardware Wizard

Windows will search for current and updated software by looking on your computer, on the hardware installation CD, or on the Windows Update Web site (with your permission).

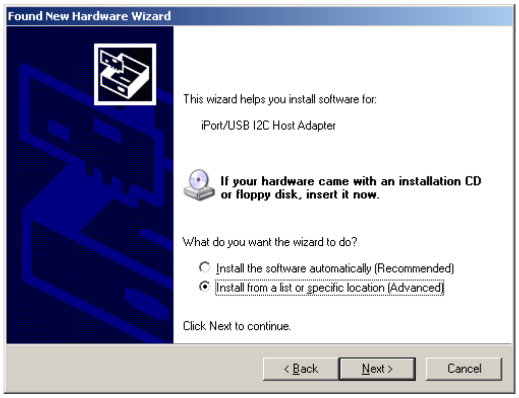
Read our privacy policy

Can Windows connect to Windows Update to search for software?

● Yes, this time only
○ Yes, now and every time I connect a device
○ No, not this time

Click Next to continue.

The next screen will ask you for directions on where to obtain the drivers from, click on *Install from a list or specific location (advanced)* and click on **< NEXT >**



This wizard helps you install software for:

iPort/USB I2C Host Adapter

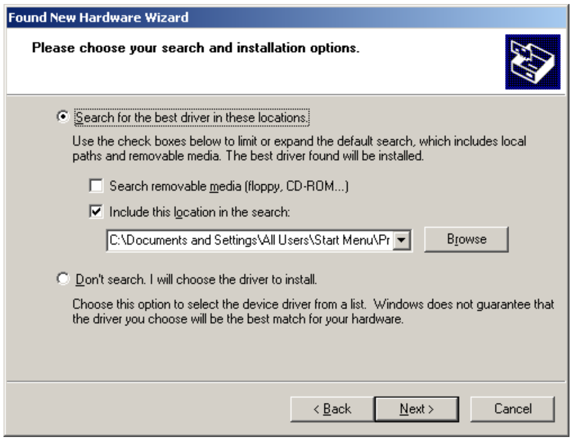
If your hardware came with an installation CD or floppy disk, insert it now.

What do you want the wizard to do?

○ Install the software automatically (Recommended)
● Install from a list or specific location (Advanced)

Click Next to continue.

Next, the installation wizard will ask where to look for the driver. Click on *Search for the best driver in these locations* and check (click) *Include this location in the search* then click **< BROWSE >** and identify the following *v4.28.0.2700* which is located at **C:\Program Files\MCC\ iPort USB\drivers\.** Note that this driver was downloaded on 2/25/2009. The name of the driver file may change. Until the driver file is properly identified the wizard will not highlight the **< NEXT >** button. When *NEXT* is properly highlighted click on it and the wizard will install the driver.



Please choose your search and installation options.

● Search for the best driver in these locations.

Use the check boxes below to limit or expand the default search, which includes local paths and removable media. The best driver found will be installed.

☐ Search removable media (floppy, CD-ROM...)
☑ Include this location in the search:

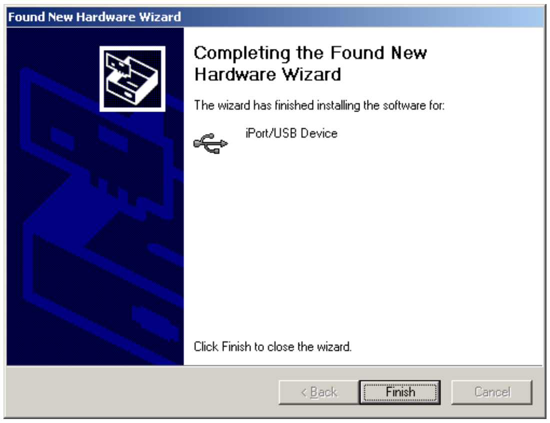
C:\Documents and Settings\All Users\Start Menu\Pr

○ Don't search. I will choose the driver to install.

Choose this option to select the device driver from a list. Windows does not guarantee that the driver you choose will be the best match for your hardware.

The path in the above wizard screen capture is incorrect. Please ignore it and follow the written directions.

When finished the wizard will acknowledge completion of the installation:



Completing the Found New Hardware Wizard

The wizard has finished installing the software for:

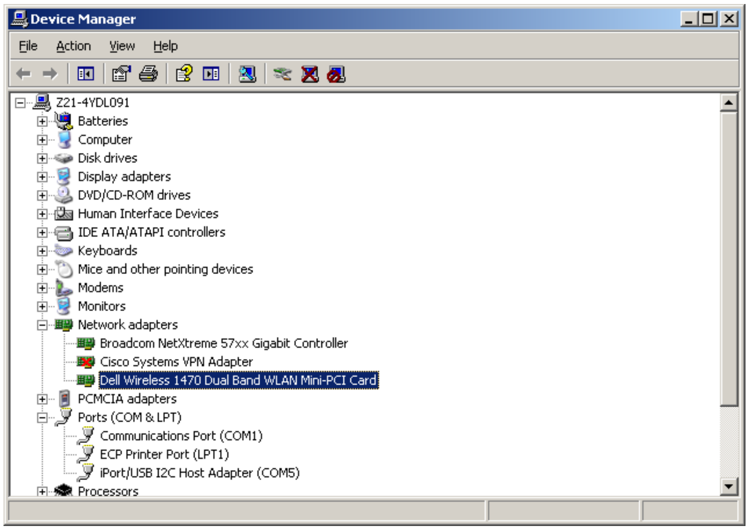
iPort/USB Device

Click Finish to close the wizard.

5.3 Identifying the communications port

The next thing you need to do is identify which communication port the USB_IPort is plugged into.

On the screen click on **< START >, Settings, Control Panel, System, Hardware, Device Manager,** and the **+** sign to the left of the **Ports (Com & LPT).** The USB_IPort should be properly identified, (see below).



- Z21-4YDL091
  - Batteries
  - Computer
  - Disk drives
  - Display adapters
  - DVD/CD-ROM drives
  - Human Interface Devices
  - IDE ATA/ATAPI controllers
  - Keyboards
  - Mice and other pointing devices
  - Modems
  - Monitors
  - Network adapters
    - Broadcom NetXtreme 57xx Gigabit Controller
    - Cisco Systems VPN Adapter
    - Dell Wireless 1470 Dual Band WLAN Mini-PCI Card
  - PCMCIA adapters
  - Ports (COM & LPT)
    - Communications Port (COM1)
    - ECP Printer Port (LPT1)
    - iPort/USB I2C Host Adapter (COM5)
  - Processors

In this case (my computer) the IPort USB adopter is present in port (COM5). If a question mark next to the IPort USB adopter appears then the computer did not identify the device. In this case right click on the **? iPort/USB..** and follow directions to install the driver again.

5.4 Configuring the communications port

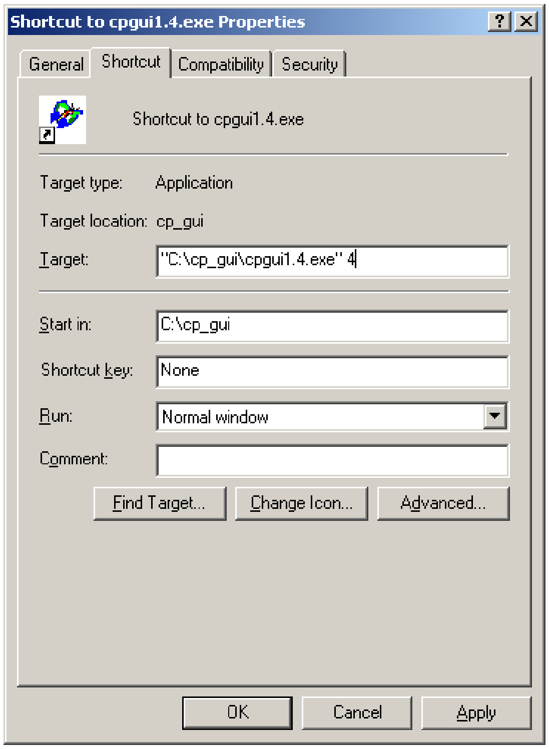
The GUI always defaults to using COM1 unless told otherwise. The communications port can be changed by modifying the GUI executable. Below are the steps for changing the communication port;

- Right click (on mouse) the executable **cpgui1.4.exe** and click on *Create Shortcut*. (On my computer, the executable is identified as **cpgui1.4.exe** and its directory is in **C:\cpgui\**) **.** A new file type shortcut will appear and it is called **Shortcut to cpgui1.4.exe.**

- Right click (on mouse) on the **Shortcut to cpgui1.4.exe** , left click on Properties. The following screen below will appear; put brackets around the command followed by a space followed by a number that is one less than the communications port that the USB_IPort is connected to. **“C;\cp_gui\cpgui1.4.exe” 4 .** (On my computer the communication port # is 5 and therefore I entered the number 4).

- Click on **< APPLY >** and **< OK >**.

- The shortcut will appear in the same directory you are in. Note that the communications port information is hidden. You can double click on the shortcut from this directory or you can move it to your desktop and start the GUI from there. The GUI should properly identify the USB_IPort as the proper communications port.



Target type: Application

Target location: cp_gui

Target: "C:\cp_gui\cpgui1.4.exe" 4

Start in: C:\cp_gui

Shortcut key: None

Run: Normal window

Comment:

5.5 Reconfiguration

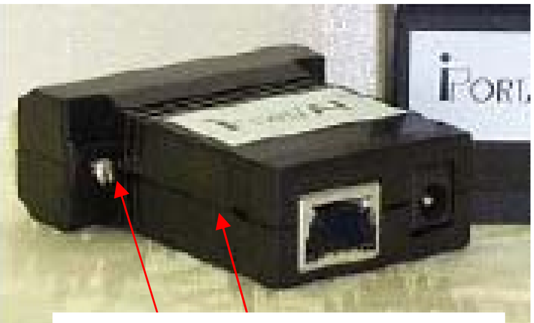
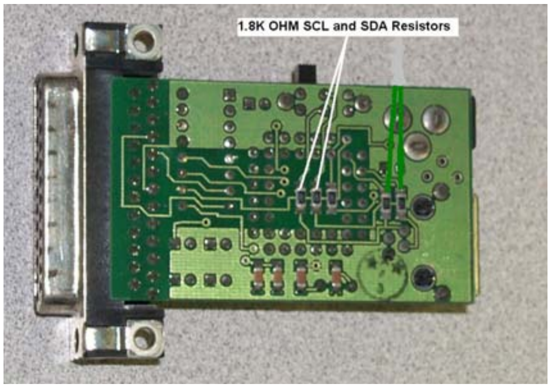
The shortcut must be changed every time the USB communications port location is changed. It is prudent to assume that the port selection has been compromised every time communications errors appear.

On one occasion we unplugged one of the IPorts and inserted another IPort in the same USB slot. The computer incremented the assigned communication port number by one. So every time a hardware connection change has been made go back to the control panel and verify that the USB slot communications port has not been changed. If it has been changed the GUI comport communications settting must be changed accordingly.

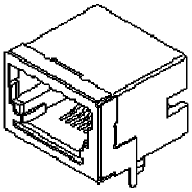
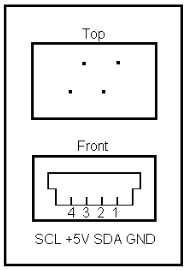
5.6 Modifying the IPort/AI pull up resistors.

The IPort/AI was designed to communicate to only a single device. The 1.8KΩ pull up resistors of clock and data of the MIIC-202 IPort should be increased to 10KΩ in order to ensure that communicating with up to 8 power supplies does not exceed the maximum current sink requirements of the i2C specification.

The picture below shows the location of the resistors. Removing the cover requires the insertion of a small screw driver into four slots, two on each side of the IPort, and turning the screw driver to unsnap the housing. Each slot can be unsnapped one at a time. The components are located on the solder side of the board. After changing the resistors reposition the components and simply snap together the base and the cover. **After modifications set the pull-up switch to the ON position.**

 Near side slots for opening the case	

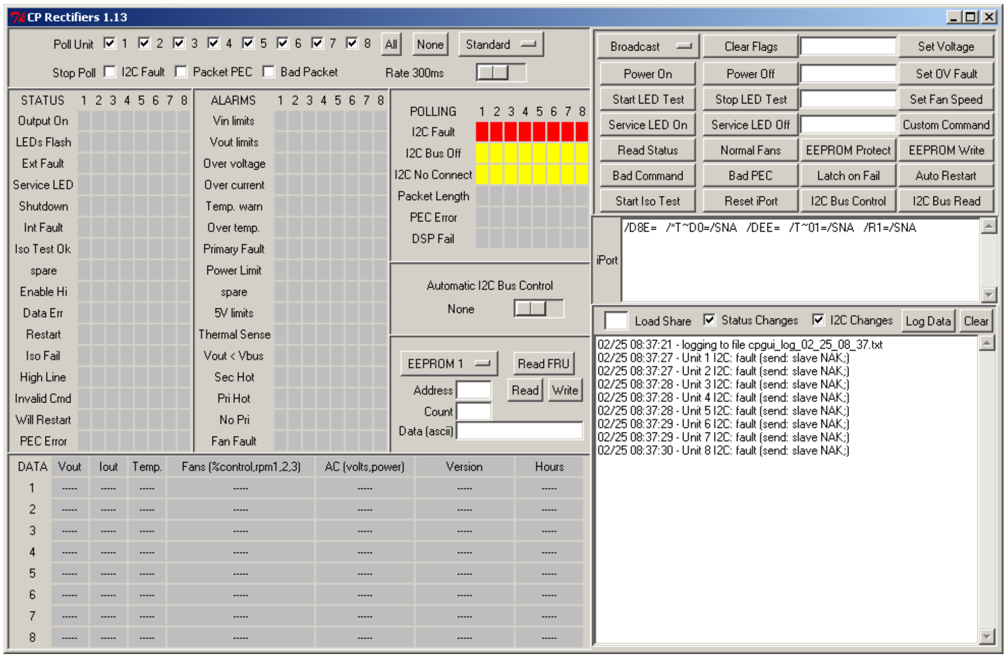
The IPort also requires a special SMBUS connector for interfacing to its output cable. The Lineage Interface Board already includes this connector. The connector can also be procured from MCC.

 Molex SEMCONN^{TM} ACCESS.bus Receptacle Connector	 **iPort I^{2}C Connector Pinout**

6 Communication errors

6.1 i^{2}C line is not connected

The IPort is connected to the computer but the i2c line is left off in error from the power system. The 'state change section' message content in the lower right hand corner annotates that the 'slave' does not acknowledge its address. Polling will show the following:



```
/D8E=   /*T~D0=/SNA   /DEE=   /T~01=/SNA   /R1=/SNA
```

```
02/25 08:37:21 - logging to file cpgui_log_02_25_08_37.txt
02/25 08:37:27 - Unit 1 I2C: fault (send: slave NAK;)
02/25 08:37:27 - Unit 2 I2C: fault (send: slave NAK;)
02/25 08:37:28 - Unit 3 I2C: fault (send: slave NAK;)
02/25 08:37:28 - Unit 4 I2C: fault (send: slave NAK;)
02/25 08:37:28 - Unit 5 I2C: fault (send: slave NAK;)
02/25 08:37:29 - Unit 6 I2C: fault (send: slave NAK;)
02/25 08:37:29 - Unit 7 I2C: fault (send: slave NAK;)
02/25 08:37:30 - Unit 8 I2C: fault (send: slave NAK;)
```

DATA	Vout	Iout	Temp.	Fans (%control,rpm1,2,3)	AC (volts,power)	Version	Hours
1	-----	-----	-----	-----	-----	-----	-----
2	-----	-----	-----	-----	-----	-----	-----
3	-----	-----	-----	-----	-----	-----	-----
4	-----	-----	-----	-----	-----	-----	-----
5	-----	-----	-----	-----	-----	-----	-----
6	-----	-----	-----	-----	-----	-----	-----
7	-----	-----	-----	-----	-----	-----	-----
8	-----	-----	-----	-----	-----	-----	-----

6.2 Read Back from a specific module fails with PEC error

The most likely reason for consistent read back PEC fault of a certain slot is an addressing problem within the modules whereby two modules may configure themselves to the same address. When two modules respond at the same time it is likely that the combined data and PEC calculation will not match.

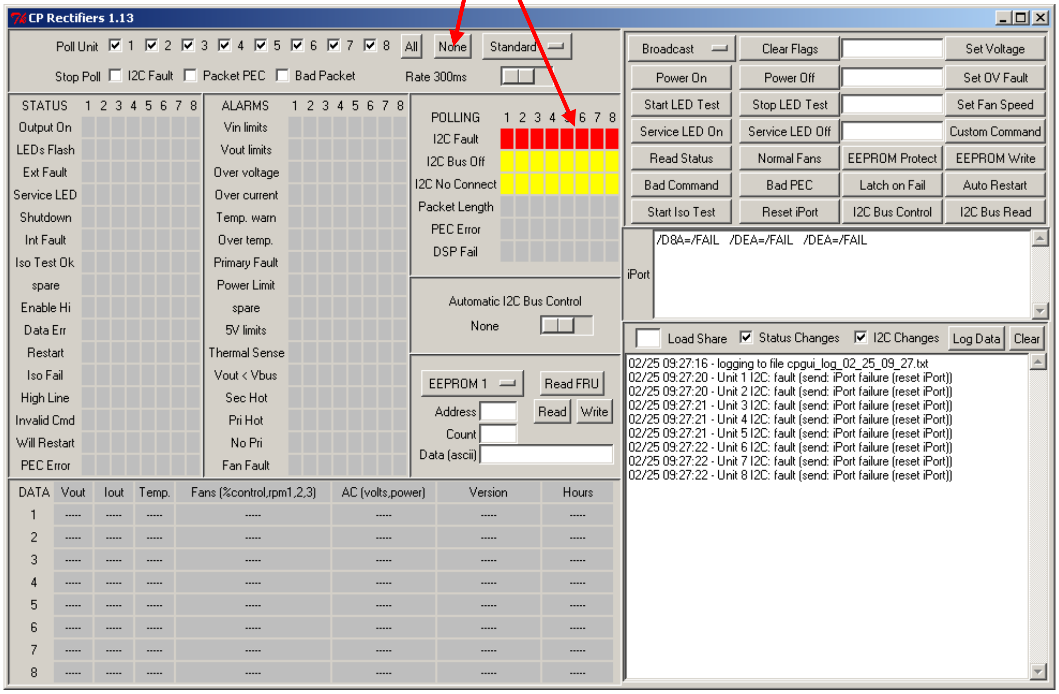
To detect whether an addressing problem has occurred issue a **Start_LED_Test** command to a specific slot and make sure that only the module in that slot responded to the command. Continue with this instruction until all the modules have been addressed. If the offending power supply is identified, unplug the power supply, wait until the fans stop spinning, and then reinsert the power supply. Go through the LED test routine again and make sure that each power supply responds only to its own address.

Configuring itself to the incorrect address could happen if a power supply was reinserted too fast and it remembered its old address. Some of the power supplies read their address only once during power up, while the latter vintage power supplies are smarter and correct their address even after insertion.

6.3 IPort missing

Either the IPort is not connected to the computer, not plugged into the correct communications port or in case of the IPort/AI, the external 5V power supply is not providing power to the IPort. In this case the ‘state change section’ message in the lower right side window indicates that the application cannot communicate to the IPort and that resetting the IPort has failed.

(Note that the ‘state of communications’ section also indicates that none of the modules are communicating. This information must have been recorded previously because if the IPort is not connected messages could not have been received. This brings up another important point, state messages are maintained until another communications state over-writes them. The only way to clear the state messages is by clicking on the **None** button in the polling section of the GUI)



DATA	Vout	Iout	Temp.	Fans (%control,rpm1,2,3)	AC (volts,power)	Version	Hours
1	-----	-----	-----	-----	-----	-----	-----
2	-----	-----	-----	-----	-----	-----	-----
3	-----	-----	-----	-----	-----	-----	-----
4	-----	-----	-----	-----	-----	-----	-----
5	-----	-----	-----	-----	-----	-----	-----
6	-----	-----	-----	-----	-----	-----	-----
7	-----	-----	-----	-----	-----	-----	-----
8	-----	-----	-----	-----	-----	-----	-----

```
/D8A=/FAIL   /DEA=/FAIL   /DEA=/FAIL
```

```
02/25 09:27:16 - logging to file cpgui_log_02_25_09_27.txt
02/25 09:27:20 - Unit 1 I2C: fault (send: iPort failure (reset iPort))
02/25 09:27:20 - Unit 2 I2C: fault (send: iPort failure (reset iPort))
02/25 09:27:21 - Unit 3 I2C: fault (send: iPort failure (reset iPort))
02/25 09:27:21 - Unit 4 I2C: fault (send: iPort failure (reset iPort))
02/25 09:27:21 - Unit 5 I2C: fault (send: iPort failure (reset iPort))
02/25 09:27:22 - Unit 6 I2C: fault (send: iPort failure (reset iPort))
02/25 09:27:22 - Unit 7 I2C: fault (send: iPort failure (reset iPort))
02/25 09:27:22 - Unit 8 I2C: fault (send: iPort failure (reset iPort))
```

6.4 GUI does not communicate

Prior to **revision** 1.15 of the GUI if the IPort hung up for any reason communications stopped. There is a button for **Reset_IPort** but the user must operate the button manually. This condition can be recognized by seeing in the ‘Transmitted Data’ section of the GUI transmission FAIL reporting. Clicking on the **Reset_IPort** button recovers the IPort.

Revision 1.15 and later should do this recovery process automatically.

6.5 GUI does not start, the execution starts and then simply vanishes

The most likely cause is that the communications port setting of the GUI and the Interface do not match and therefore the GUI does not execute properly. Check the com port assignment of the IPort and the port assignment of the GUI.

Europe, Middle-East and Africa Headquarters
Lineage Power (UK
Tel: +44 (0) 1344 469 300, Fax: +44 (0) 1344 469 301

Caribbean-Latin America-Brazil Headquarters
Lineage Power
Tel: +56 2 209 8211, Fax: +56 2 223 1477

Asia-Pacific Headquarters
Lineage Power Singapore
Tel: +65 6416 4283, Fax: +66 6416 4299

India
Lineage Power India
Tel: +91 80 841 1633 x3001

World Wide Headquarters

Lineage Power
3000 Skyline Drive, Mesquite, TX 75149, USA
+1-800-843-1797
power.lineagepower.com

Lineage Power reserves the right to make changes to the product(s) or information contained herein without notice. No liability is assumed as a result of their use or application. No rights under any patent accompany the sale of any such product(s) or information.

April 2009
GGS Rev 1.5