EVILbus Specification

Last Updated: 11/18/05 6:17 PM

Author: John Lussmyer
Contributors: Lee Hart, Ralph Merwin, Roger Stockton

Table of Contents

Tables

1 Introduction

1.1 What is the “EVILbus”?

It's an inside joke between Jon Pullen and I (Lee Hart). We started with "Electric Vehicle bus by Jon and Lee", but E V J L didn't spell anything. I said it's too bad his name isn't Igor. Jon suggested Electric Vehicle Instrumentation bus by Lee (EVILbus).
It is actually the bus used to communicate between the individual battery chargers in Rick Woodbury's Tango prototype.

1.2 Description

- a 2-wire open-collector optically-isolated bus
- 32 nodes in present configuration; 3-4 times this number is feasible
- idle bus power consumption is 0.5ma per node
- Connector: RCA phono plug
  The Tango chargers have a 3-foot "extension" cord with a male RCA on one end, and a female RCA on the other. I cut it in the middle and solder it to the PC board; Thus each charger has two 18" cables, male and female, making it easy to daisy-chain them all together. - center pin positive, outer shell negative
  Why? common, inexpensive, shielded cable widely available, connections are easy to seal up with a piece of heat shrink tubing over the connection

2 Electrical Specification

2.1 Logic Levels

- >9v is the idle or "1" or "stop bit" state
- <1.5v is the active or "0" or "start bit" state

2.2 Termination

- 150 ohm resistor from the center pin (positive) to +12v
- 150 ohm resistor from outer shell (negative) to GND
- 12v supply can be anything from 9v to 16v
- Bus has only ONE set of terminators.
- Why? they serve as pull-up resistors, and terminate noise and ringing on the bus. Since the bus is used at low speeds, these resistors can be located anywhere on the bus. Having the resistance split between the + and - sides means a short circuit either does nothing, or at worst blows out one of these resistors which are easy to replace.

2.3 Receiver

- 10k ohm resistor (R6) and LED of an optocoupler (IC1) in series
- series blocking diode (D1) so reverse polarity does no harm
- 10k ohm resistor (R3) across LED insures leakage current won't falsely turn LED on
- input current in passive high state (mark): 410uA at 9v, 590uA at 12v, 810uA at 16v. Multiply by number of nodes for total 12v supply current
- input impedance approximately 12k ohms
- input "high" threshold is approximately 400uA at 5v
- Input voltage in active low state (space): 2.2v @ 16mA (9v), 2.3v @ 30mA (12v), 2.4v @ 45mA (16v) current limited by 150+150 ohm pullup resistors
- Why? Low input current is needed to conserve power, but low-current optocouplers are slow. Optocoupler (IC1) shorts base-emitter junction of driver transistor (T1) to improve speed.

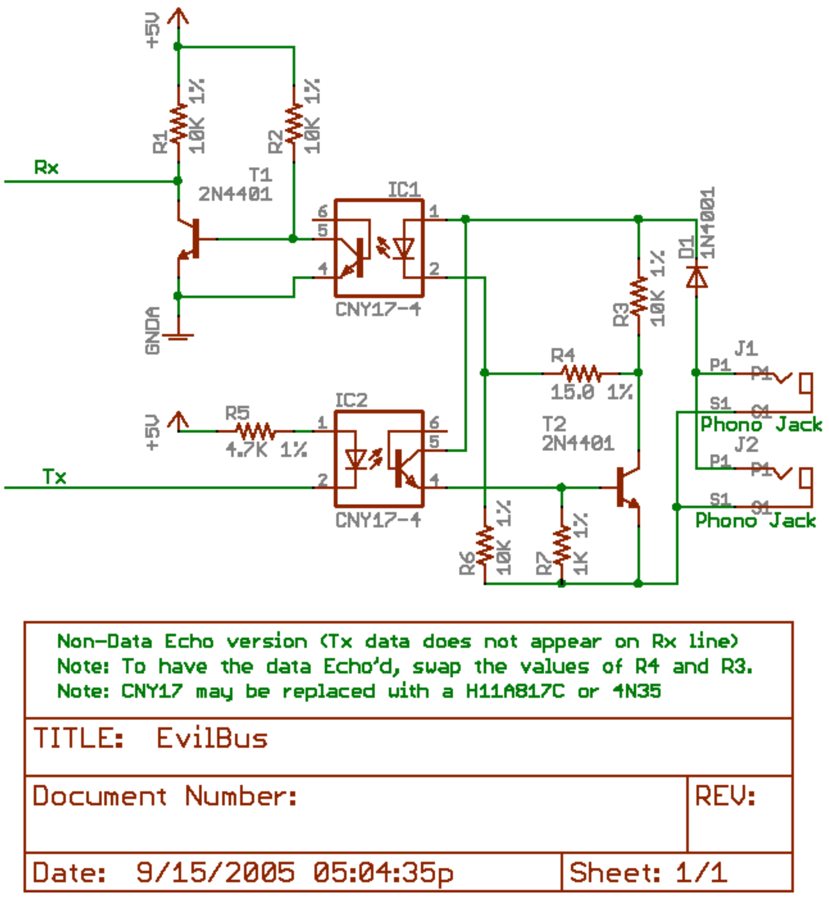
2.4 Transmitter

- Optocoupler (IC2) phototransistor driving base of transistor (T2) to short the two wires together
- Why? logic driving the optocoupler need only supply 1ma to its LED, yet 10's of ma can be carried by the transistor. Didn't use a darlington optocoupler because they are slow.

2.5 Isolation

- 2500 volts (set by the optocouplers I used)

2.6 Sample Schematic



Non-Data Echo version (Tx data does not appear on Rx line)
Note: To have the data Echo'd, swap the values of R4 and R3.
Note: CNY17 may be replaced with a H11A817C or 4N35

TITLE: EvilBus

Document Number: REV:

Date: 9/15/2005 05:04:35p Sheet: 1/1

Photo 1

3 Bus Information

3.1 Detecting Bus Condition

- a continuous "0" or start bit means there is no EVILbus connected, or it is unpowered. (or a Long Break is being transmitted.)
- a continuous "1" or stop bit means the EVILbus is connected, but there is no data at present.

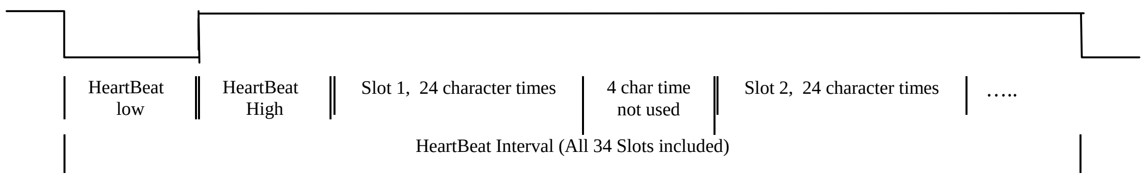
3.2 Serial Data Format

9600 baud, 7 Data Bits, Even Parity, One stop bit.

At 9600 baud, there is time for 960 characters of data to be transmitted in 1 second. We use 3 of these for the Heartbeat signal, so that leaves time for 957 characters of data – if the bus was perfectly utilized, which isn't likely to occur in the real world.

3.3 Heartbeat

This bus uses a Time Domain Multiplexing communication system. In order to allow all the various nodes to synchronize with each other, there will be a "Heartbeat" signal generated on the bus to indicate the start of the standard time interval.
This "Heartbeat" is indicated by the bus going low (0) for 4 character times at 9600 baud, then high for the next 4 character times. (Approximately 4 milliseconds low, then 4ms high.) This will happen once every Heartbeat Interval. (The default Heartbeat Interval is 1.0000 second.) The time between heartbeats is when normal device data communications happen. Each device has a "Slot" during the Heartbeat Interval in which to send it's data.


HeartBeat low
HeartBeat High
Slot 1, 24 character times
4 char time not used
Slot 2, 24 character times
…..
HeartBeat Interval (All 34 Slots included)

The Heartbeat should be used by nodes to adjust their timing so they stay in sync with other nodes.

3.4 Heartbeat Generation

The Heartbeat may be generated by any device on the bus, but any device doing so must be able to synchronize itself with any other device that may be generating the heartbeat. In general, only devices that consume data need to be able to generate a Heartbeat.
Devices that only provide data, and don't consume it, don't need to be able to generate a heartbeat – as there is no point in providing data if nobody is listening to it.
It is suggested that any device that wants to generate a heartbeat should listen to the bus first, and only generate a heartbeat if one isn't already present. Note that this may be used by some devices to "take control" of the heartbeat generation if they have a special reason to do so. This would be by generating a "Long Break" (see later description) and then generating the heartbeat so that they have control over it.

3.5 Slot Timing

The default Slot duration for a node is 28 character times. This is followed by a 4 character time during which the node must not send data, this is used to keep nodes that are slightly out of sync with each other from stepping on another nodes data. There are commands to allow non-default timings to be used.

3.6 Long Break

A Long Break is generated by pulling the bus low for a full Heartbeat Interval. This does not have to start at a Heartbeat, just last for a full Heartbeat Duration of milliseconds.
A Long Break is used to interrupt the normal bus operation so that commands may be sent to nodes. This is done so that nodes don't have to be listening to the bus at all times to watch for commands, and to prevent collisions caused by multiple devices sending commands at the same time.
The device that generated the Long Break has control of the bus, and all other devices should listen to the bus and NOT send their data packets. This condition continues until:

a) The bus has been idle (no communications) for 30 seconds
b) A Heartbeat happens

After a long break is when Commands may be sent on the bus, and only the device the command is addressed to may reply to it.

4 Communication Protocol

4.1 Node ID

Device Node ID's shall be in the range of 1 to 99.
Nodes with ID's over 32 are reserved for devices that don't normally need to send data during the normal data time interval. These are generally devices that consume data for display or other control reasons.
Node ID 99 is reserved for new nodes that have not yet been configured. Note that this means that only 1 new node may be added to a bus at a time to prevent ID conflicts.

4.2 Node Types

There are 2 general types of nodes. Data Producers and Data Consumers. It is possible for a Node on the bus to be both types at once.

4.2.1 Data Producers

These are nodes who's primary purpose is to put data on the bus for other nodes to use. These need a Node ID that lets them send their data between Heartbeats.

4.2.2 Data Consumers

These are nodes who's primary purpose is to make use of the data on the bus. They generally (but not always) can use a Node ID that doesn't allow data to be sent between Heartbeats.
They are likely to be able to produce a Heartbeat if one is needed. They may also use the Long Break system to send commands to other nodes.

4.3 Communication timing

In order to prevent collisions on the bus, each Data Producer is assigned a portion of the time between Heartbeats when it may send data on the bus. There are 2 ways of assigning this time interval, Basic and Advanced. Timing for either one starts on the trailing/rising edge of the Heartbeat signal.

4.3.1 Basic (Default) Timing

Devices should default to this version of Message Timing. For this system, each Node is assigned a Timeslot based on it's Node ID. This timeslot is 28 characters long, but the node may only use the first 24 characters of time to send data. The remaining 4 are a required "quiet" time for the bus.
The node must NEVER send data outside of it's Time Slot.

4.3.2 Advanced Timing

Devices may be configured to use an alternate message timing system that allows the bus to be more fully utilized. This is only needed in the fairly rare cases where the devices need to send more data than fits in their timeslot, or if there are special reasons to pack the data messages more closely together. This system allows each device to be assigned a specific Character time after the heartbeat to start it's data message, and a specific number of character times it may use. This is covered in more detail in the S(lot) and N(extSlot) commands. See section 6 for the full descriptions.

5 Data Packets

A Data Packet is the text that a node sends during it's normal time interval as indicated by it's Node ID.
The general packet format is required, but the details of the data within it are guidelines that new devices should try to extend in compatible ways.
Note that a device may send multiple packets during it's timeslot, as long as it doesn't exceed the timeslot duration.

5.1 Packet Syntax

5.1.1 Multiple Value, Single Type

When multiple same-type values are being reported on a single line:

{Data For}{Which}{Type}={Value}[,{Value}[,etc…]]{\n}
Example: B3V=12.5,11.6,12.3

5.1.2 Multiple Value, Multiple Type

When multiple values are being reported with different types for each.

{Data For}{Which}={Value}{Type}[,{Value}{Type}[,etc…]]{\n}
Example: B3=12.4V,44C

5.1.3 Definitions

Where:

{Data For}	Required: What kind of item this data is about. A single Alphabetic character. (See 2.2)
{Which}	Required: Which item of that type this is about, this must be an integer with 1 to 3 digits. For a multiple item device, this is the index of the first item this device is handling. Set for a node via the "W" command.
{Type}	Data value type this message contains. A single Alphabetic character. (See 2.3)
{Value}	Required: The value being reported, multiple values may be reported and are comma separated. Generally numeric, may be Integer or Real number.
{\n}	Packet terminator character. (LineFeed character by default)

Note that the above formats may NOT be mixed on a single line. You may either put the {Type} before the "=", or at the end of each {Value}, NOT BOTH.

5.2 Data For Kinds

The pre-defined item types are: (Must be Upper Case!)

{DataFor}	Name
B	Battery
C	Charger
N	Controller
P	Pack
L	AC Power Line
M	Meter
S	Switch

Table 1 – Data Message "Data For" Kinds

If a node (Consumer) receives a packet with an unknown Kind, it should just ignore the message.

5.3 Value Types

The pre-defined data value types are: (Must be Upper Case!)

{Type}	Unit
V	Volts
A	Amps
C	Degrees C
F	Degrees F
H	Amp Hours
W	Watt Hours

Table 2 – Message Data Value Types

If a node (Consumer) receives a packet with an unknown Value Type, it should ignore that Value.

5.4 Example Data Packets

From a single Battery Regulator/Monitor for Battery 1.

```
B1V=12.50<\n>B1C=33.4<\n>
or
B1=12.50V,33.4C<\n>
```

From a Quad Battery Regulator/Monitor for Batteries 2-5

```
B2V=12.50,11.99,12.34,12.01<\n>
B2C=33.1C,34.0,38.0,35.5<\n>
```

From a Whole Pack Monitor

```
P1V=235.0<\n>P1A=44.3<\n>
or
P1=235.0V,44.3A<\n>
```

From one of several Battery Box Temperature Sensors

```
P3C=30<\n>
```

6 Command Packets

6.1 Packet Syntax

{Cmd}{Node ID}[=Value]{\n}

Where:

{Cmd}	Required: single command letter. (see below)
{Node ID}	Required: Which Node this command is for.
[=Value]	Optional: Value for Command parameter. Varies by Command and Device.
{\n}	Linefeed Command terminator character.

6.2 Commands

These are just the set of commands that all nodes should implement. Nodes may implement private commands via the Command command to do node specific things. All commands must be lower case. In general, commands that take parameters may be used without the parameters to query the current setting.

Cmd	Name	Description
i	ID	*Required*: Sets a new Node ID for the device. The device should respond with: Ex: <cmd>i99=14<\n> set new node id <resp>ID=14,Which=14,Len=12,Slot=392,Next=404<\n> Ex: <cmd>i14<\n> Query current setting <resp>ID=14,Which=14,Len=12,Slot=392,Next=404<\n> Note: This WILL change the Nodes Slot and Which settings to the default values base on node ID. Note: The Slot Length reported must include the 4 character quiet time at the end of the nodes message.
s	Slot	*Required*: Starting time for Timeslot. The value is which character time the device may start transmitting it's data message. For normal/default Heartbeat timing, this is in the 1 to 956 range. The max for this is whatever the Heartbeat Interval has been set to. (though starting with only 1 char time before the next Heartbeat is likely to cause problems.) The Response to this command should be the new Start setting, and the next available Start time for another device. Ex: <cmd>s11=23<\n> Set slot number to 23 <resp>ID=14,Which=14,Len=17,Slot=23,Next=44<\n> Ex: <cmd>s11<\n> Query current setting <resp>ID=14,Which=14,Len=17,Slot=23,Next=44<\n> Note: May NOT be sent as a Node 0 command. Note: The Slot Length reported must include the 4 character quiet time at the end of the nodes message.

n	Next Slot	*Required*: Specifies what the starting time of the following device will be. The difference between this and the Starting Slot is how many characters may be sent by this device. The device should respond with how much of that time it will actually use, and what the actual Next available timeslot is. The node isn't required to use the full number of slots provided. If the value is less than the nodes minimum message length, it should return an error message. Ex: `<cmd>n24=300<\n>` Set Nextslot to 300 `<resp>ID=24,Which=17,Len=17,Slot=56,Next=73<\n>` Ex: `<cmd>n24<\n>` `<resp>ID=24,Which=17,Len=17,Slot=56,Next=73<\n>` Note: May NOT be sent as a Node 0 command. Note: The Slot Length reported must include the 4 character quiet time at the end of the nodes message. Note: It is suggested to set the Which value before setting the Next Slot value.
w	Which	*Required*: Set the index for the item type this device is reporting for. Value must be an integer. If given without a value, it just queries the current setting. Default Value: The device Node ID Device should respond with the number of items it covers, and what the next available item number is: Num={Number of items},Next={Next available Item number} Example for a 4 battery regulator: Ex: `<cmd>W3=5<\n>` Set Which item to 5 `<resp>Which=5-8<\n>` Shows it uses numbers 5 through 8 Ex: `<cmd>W3<\n>` Query which item number `<resp>Which=5-8<\n>` Note: May NOT be sent as a Node 0 command Note: It is strongly suggested that the user set the Which value before setting the Next Slot. There are some nodes that may change the length of their Slot depending on how many digits are in the Which number..
t	Talk	*Required*: This MUST be broadcast as a Node 0 command. The value specified is the ID of the node that is going to take full control of the bus. All other nodes should completely ignore the data on the bus until after the next Heartbeat or Long Break. All 30 second bus-idle timers should also be disabled. Even if a node never needs full bus control, it must at least listen for this command, so it knows to ignore the data bus until the next Heartbeat.
!	Info	*Optional*: Node should respond with it's identity and current settings This is the required format: {Make},{Model},{Ver=xxx},{ID=Node ID},{Min=Min Message Packet Length},{Max=Max Message Packet Length},{Cmds=Supported Command Characters} Ex: `<cmd>!3<\n>` `<resp>Whizzer,Single Reg,Ver=1.22,ID=3,Min=8,Max=44, Cmds=IQT!?DWSNS`
?	Help	*Optional*: Node should respond with help about what commands it supports and what they do.
d	Data	*Optional*: The node should respond with it's normal Data Message.
h	Heartbeat Interval	*Optional*: Sets the heartbeat interval in milliseconds. This is provided for unusual cases where the 1 second heartbeat is not appropriate. The parameter is the Heartbeat Period in milliseconds. The normal default Heartbeat is 1000. Allowed range is 50 to 30,000 milliseconds. When setting a new Interval, it may ONLY be sent to Node 0, as ALL nodes must synchronize with the new Heartbeat when it starts. Ex: `<cmd>h0=700<\n>` sets Heartbeat to 700 milliseconds No response If you are just querying the current setting, you may send this to a specific node,

		with no value given. Ex: <cmd>h12<\n> <resp>Heartbeat=1000<\n> <u>Note</u>: If a node uses the length of the Heartbeat Interval to adjust it's internal clock, then it must listen for this command so that it knows when the interval has been changed.
c	Command	*Optional*: Indicates a Node Specific command is being given. The "Value" for this is whatever the Device has indicated is allowed. Ex: <cmd>c12=ti=40<\n> <resp>TempInt=40<\n> This could be for Device 12 to set the Temperature Interval to 40 seconds. <u>Note</u>: Since this may have a 2nd '=' in it, the private command must NOT look like a valid general command.
b	Beat	*Optional*: Start the Heartbeat again. This would only be used if a device that doesn't normally produce a heartbeat, and may not be capable of doing so, had control of the bus via Long Break. It may then broadcast this command and the Device that was last producing the Heartbeat should resume doing so. <u>Note</u>: It is strongly suggested that Data Consumers that are able to produce a Heartbeat implement this command. <u>Note</u>: If a device is normally able to produce a Heartbeat, receives a Beat command that is addressed directly to it, it should start producing a Heartbeat. If the Beat command was broadcast to node 0, and it wasn't producing the Heartbeat before the Long Break, then it should allow the previously beating device to produce the Heartbeat. If no other device does so within a reasonable interval (2 * (Heartbeat interval + ID))ms then it should initiate the Heartbeat. Ex: <cmd>b12<\n> Tell node 12 to start producing a Heartbeat. <resp>Hearbeat=1000<\n> Response only because the Beat command was addressed to this node specifically. The node could respond with an error message as well.

Table 3 – Device Commands

6.3 Command Responses

If a Node receives an Unknown (or Unsupported) Command that is addressed directly to it, it should respond with some kind of error message.
All command responses, including those of Private Node Specific commands, must be formatted so that they do NOT look like a valid command to another node.

6.4 Node ID 0 (Broadcast)

If a command is sent to a Node ID of 0, that means that it should be executed by ALL nodes on the bus. No response to the command is allowed. (This means it is silly to send a query command (like Info) as a broadcast.)
This does NOT terminate the Long Break command session, it is just a way to get data from all nodes with a single command.
If the command is unknown or unsupported by the node, it should quietly ignore it. Note that for the Heartbeat Interval command, this may or may not cause problems.

6.5 Node ID 99 (New Node)

Node ID 99 is reserved for new, not initialized nodes. Nodes with an ID of 99 should only respond to the ID, Info, and Help commands. It should also assume that the Quiet command was already given to it.

6.6 Examples

6.6.1 Basic Bus Setup

Here is an example of setting up a new bus of devices using the default basic timing system.
We have 4 devices to install. 2 Quad Battery monitors, a Pack monitor, and a Data Display.
We will use a terminal to do the setup.

- Connect a Quad Battery Monitor to the bus.
- *{Cmd}* i99=2{enter}
- *{Response}* ID=2, Which=2,Len=21,Slot=28,Next=49
- Connect the other Quad Battery Monitor to the bus.
- *{Cmd}* i99=3{enter}
- *{Response}* ID=3, Which=3,Len=21,Slot=56,Next=77
- Connect the Pack Monitor to the bus.
- *{Cmd}* i99=4{enter}
- *{Response}* ID=4, Which=4,Len=15,Slot=84,Next=99
- Connect the Data Display unit to the bus.
- *{Cmd}* i99=5{enter}
- *{Response}* ID=5, Which=5,Len=0,Slot=112,Next=112

At this point the user may just wait for the 30 second timeout, or use the Beat command to start the Heartbeat running.

6.6.2 Advanced Bus Setup

Here is an example of setting up a new bus of devices using the advanced timing system. This may be done because the user is planning on adding lots of devices that produce a lot of data.
We have 4 devices to install. 2 Quad Battery monitors, a Pack monitor, and a Data Display.
We will use a terminal to do the setup.

- Connect a Quad Battery Monitor to the bus.
- *{Cmd}* i99=2{enter}
- *{Response}* ID=2, Which=2,Len=21,Slot=28,Next=49
- *{Cmd}* s2=2{enter}
- *{Response}* ID=2, Which=2,Len=21,Slot=2,Next=23 note that the device is using a 21 char msg since that is what it could fit in the default size of 24 characters. It may be able to use more than that though.
- *{Cmd}* n2=500{enter} note the use of an arbitrarily large number indicating that I want the node to be as verbose as possible.
- *{Response}* ID=2, Which=2,Len=35,Slot=2,Next=37 It appears this node will produce 35 characters at most, so the next available timeslot for another device is 37.
- Connect the other Quad Battery Monitor to the bus.
- *{Cmd}* i99=3{enter}
- *{Response}* ID=3, Which=3,Len=21,Slot=56,Next=77
- *{Cmd}* s3=37{enter}
- *{Response}* ID=3, Which=3,Len=21,Slot=37,Next=58
- *{Cmd}* n2=72{enter} since I know it will only use 35 characters, I just specify the correct next slot time.
- *{Response}* ID=3, Which=3,Len=35,Slot=37,Next=72
- Connect the Pack Monitor to the bus.
- *{Cmd}* i99=4{enter}
- *{Response}* ID=4, Which=4,Len=15,Slot=84,Next=99
- *{Cmd}* s4=72{enter}
- *{Response}* ID=4, Which=4,Len=15,Slot=72,Next=87 note that the device is using a 15 char msg since that is what it could fit in the default size of 24 characters. It may be able to use more than that though.
- *{Cmd}* n4=500{enter}
- *{Response}* ID=4, Which=4,Len=15,Slot=72,Next=87 It appears this node will produce 15 characters at most, so the next available timeslot for another device is still 87.
- Connect the Data Display unit to the bus.
- *{Cmd}* i99=5{enter}
- *{Response}* ID=5, Which=5,Len=0,Slot=112,Next=112
- *{Cmd}* s5=87{enter}
- *{Response}* ID=5, Which=5,Len=0,Slot=87,Next=87 note that this device apparently does not produce any data messages normally, so the timeslot is still available for another device.

At this point the user may just wait for the 30 second timeout, or use the Beat command to start the Heartbeat running.

6.6.3 Common Bus Setup

While the above gets things setup enough to run, the Display unit will be somewhat confused as the Quad Battery Monitors will be reporting that they are monitoring batteries 2 to 5, and 3 to 6. So, we may want to issue more commands:

- *{Cmd}* w2=1{enter}
- *{Response}* Which=1-4
- *{Cmd}* w3=5{enter}
- *{Response}* Which=5-8
- *{Cmd}* *w*4=1{enter}
- *{Response}* Which=1

Now the first Quad monitor will be reporting for batteries 1-4, and the the 2nd Quad Battery Monitor will be reporting on batteries 5-8, and the Pack monitor is for Pack 1.

Now, if we just leave the bus idle (don't type anything at the terminal) for 30 seconds, it will start operating normally. (Or we could reset the Display device, or the Display device may have a "start running now" command.) While the bus is running normally, the terminal will see messages like this:

```
B1V=12.4,12.3,12.5,12.4
B1C=33,34,33,33
B5V=12.5,12.4,12.4,12.5
B5C=34,33,34,35
P1V=99.1
P1A=0
```

This will repeat every second. The Battery monitors are reporting the voltage and temperature of each battery, and the Pack monitor is reporting the total pack voltage, and the current being drawn from it. Notice that the Pack voltage is not the same as the sum of the battery voltages. This is due to variations in the accuracy of the various measurements.

6.6.4 Bandwidth Conservation

Some devices may have data that doesn't need to be reported every second. One way of dealing with this is to only send this data occasionally, instead of any other data the node may be providing.
Example: Quad Battery Monitor node
Normal Message

```
B4V=12.14,12.05,12.11,12.08
```

Then, every 60th message (i.e once a minute) instead of the Voltage, you get

```
B4C=33,34,34,35
```

Which are the temperature measurements.
This is just a possible way for a node to preserve bandwidth if it might be necessary. It is suggested that any such optimization be controlled by a node-specific command.