Scale Rail Article

Introduction:
The advantages of a bus: reduced wiring, ease of automation, adding a computer. A PD solution would be good, and get everyone onto the same page.

History:
DIDMetaDCC --> MRLCB --> NMRAnet
--> CBUS -->Atbus, -->S9.5, -->S9.6
Thanks to the many contributors: ...

Similarities:
CAN
Producer-consumer

How S9.6 is different:
We have thought about the difficult problems:
how to make it easy for the novice
generalizing the p/c model
simplifying the simple nodes
supporting the very large layouts

Simplicity for the novice, and the expert alike
- no need to learn Ids, channels, etc
- push button programming
- useful things: automating a station's tracks
- automating a yard throat

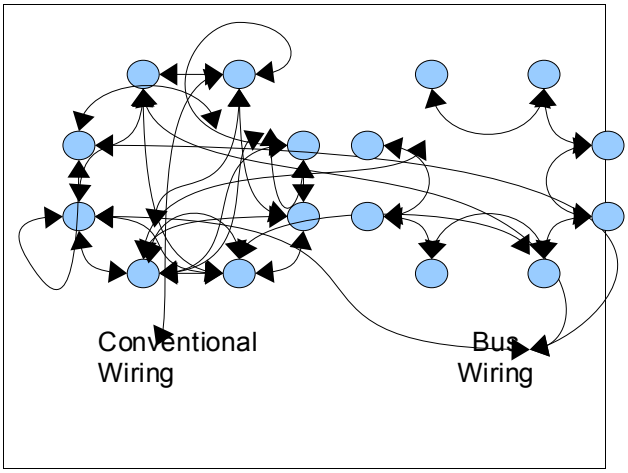
Generalizing the P/C model-
- any node that shares an event# responds to the event
- any node can be taught the event from any other node

Large Ids simplify things:
- allows us to fold other information into the events
- this lets any 'dumb' node respond to the event
  eg: fast clock. n S9.5, this would be a new set of messages, to use them, each node woul dhave to have code added to handle the new set of messages. With S9.6, because of the big-events, the time is folded into the events. A dumb node can respond to the time code, just like any other event. It doesn't need reprogramming.

Support for very large layouts:
- If your bus gets full, either too many nodes, or too much traffic – just split it in two.
- A large layout can be a number of segments joined by a fast bus.
- Traffic is reduced by gateway filtering, messages only go where they are wanted.
- This is called “scaling gracefully”.

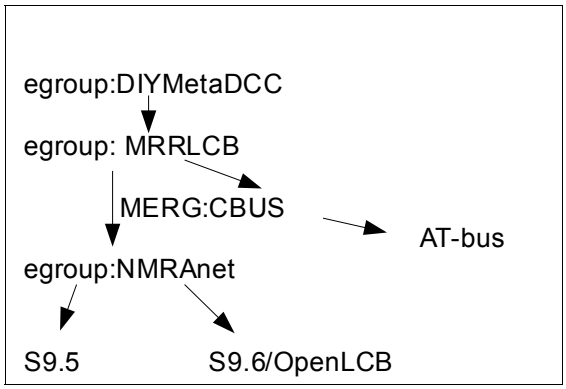
Introduction

Although NMRA has had great success with the standardization of DCC, the rapid advance of electronics has meant that the control of accessories, animation and ... on the layout is much easier. However, the wiring of all these accessories leads to a spider-web under the layout. Unfortunately, one solution that many groups have tried, that of using DCC as an accessory bus, has lead to DCC congestion. Alternate methods include using one of the proprietary CAB or throttle-buses, such as Loconet, Xpressnet ar NCEs Cab-bus. The NMRA decided that a new public domain accessory bus should be developed. [picture of ad hoc wiring vs bus]



History

Numerous people have taken part in a series of internet egroups, some predating the NMRAs interest in the bus. The first group, DIYMetaDIY, was started by Chuck Card. The discussion moved to MRRLCB, and eventually to NMRAnet, when the NMRA got involved. There was some consensus: CAN was recognized as an ideal bus for MRR, having been used in cars ans trucks, as it is robust, error strong. The pc model was introduced to the group by the ACCBUS group, Don Voss and John Socha-Leialoha, and was recognized as a powerful method that both simplified and made the bus protocol more powerful. The severe space-limitations of CAN packets lead to disagreements on how to design the packets. Two members were eager to build a bus, and split off to develop the CBUS at MERG, based on 11-bit headers. Subsequently, another MERG member, Roger Edwards modified the protocol and developed AT-bus.



The rest of the group continued to discuss the use of CAN but focusing on 29-bit headers. This group eventually split into the S9.5 effort, lead by Don Voss, and the S9.6 effort comprising John Day. David Harris, Bob Jacobsen, Alex Shepherd, and Alan Robinson. Di Voss has done a good job introducing the concepts of an accessory bus and the producer-consumer model, and the S9.5 proposal. This article will concentrate on describing the advantages of S9.6.

How S9.6 Differs

The S9.6 group wanted to design a bus standard that added significant functionality to that provided by existing buses, and one that one that will be useful into the next 10-20 years. For this reason the S9.6 design team spent a lot of time thinking about the hard problems: how to make it simple for the novice and small layouts, but let it serve very large layouts; and how to let it take advantage of future technology while simplifying the implementation of individual nodes.

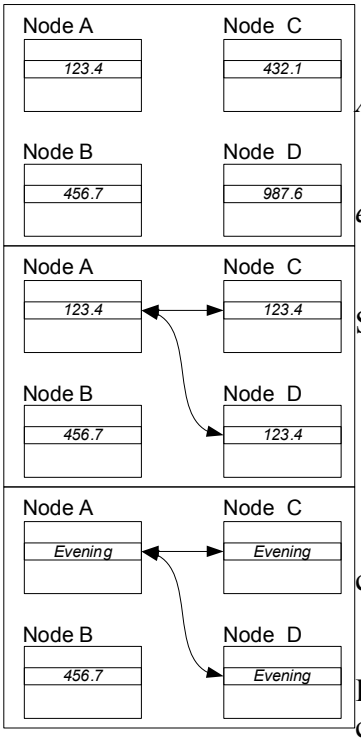
Support for the novice:
- plug and play
- no node, channel, event numbers to worry about
- push-button programming
- no id conflicts

Support for large layouts:
- no id conflicts, bring together modules set up in isolation
- no need to keep extensive lists of node and event numbers
- auto-topology and tools to configure a meet, drill-down capability

Scalability:
- multiple segment supprt with high speed backbone
- automatic traffic management and routing

Adaptablility / Support for legacy equipment
- other protocols folded into the event-space
- other protocol ids folded into node-space

Three base protocols: events, datagrams, and streams.



Easy for the novice

Alice wants to control her three track station. She has goes to her local store and buys one 8-button input node, four turnout-controller nodes, and a power supply. When she gets home, she unpacks the nodes and connects each of the turnout-controllers to her four turnouts. She connects the nodes together, teaches the buttons to control her turnouts.

S9.6 lets one node teach its event numbers to other nodes. In the upper diagram to the left, the four nodes are represented by boxes containing an event number. Since they do not share any event numbers, they cannot communicate.

In the middle diagram, node A has taught its event number to nodes C and D and formed a relatinoship or *event*. This event can represent any conceptual idea, such as 'evening has fallen' – this is shown symbolically in the bottom panel.

In S9.6 the teaching can be done by a series of button presses, and this is called the '*blue/gold*' teacing algorithm.

Alice does not need to set any node, channel, event numbers – these were pre-defined at the factory. She can plug-and=play with her nodes.

Big Events

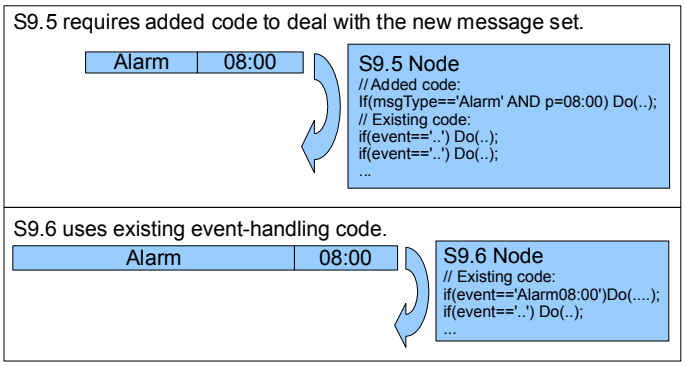
Your club does not want to expend any effort pre-allocating numbers, making sure that module configurations don't conflict or anything like that; they just want to plug them together and operate the entire layout from both central and distributed locations. All the modules must still continue to operate without reprogramming or reconfiguration; there's just not enough time for that.

A major difference from previous buses, and the other proposals, is the use of 'big events' - S9.6 uses 48-bit and 64-bit node-ids and event numbers, respectively. This decision was not taken lightly. Big events have many advantages for you. It simplifies the code of simple nodes, and increases flexibility since all nodes can respond to any event. It frees you from remembering node, channel and event numbers since these are pre-assigned at the factory, and each is unique. Since the events are unique, your modular club and other large layouts won't running out of node numbers and won't hav e any conflicting nodes.

As an example of simplifying nodes while adding functionality, consider adding a fast clock with an alarm function to your layout. The old way would to be to define a new alarm-messages, and to write new code into each node to recognize and process those new messages.

Big-events let S9.6 enclose the alarm time-value right inside an event. All your nodes can use it just like any other event. There is no need to add code to the node to add this function.

This is illustrated at the right. The top panel shows how the old way would add the alarm-event and the extra code required to recognize and process it. The bottom panel shows S9.6's approach -- the time is enclosed in a regular event, and the node responds to it just like any other event.

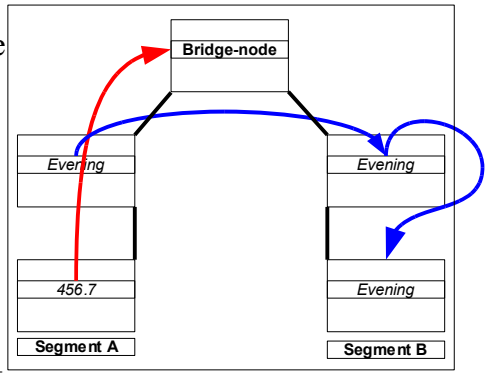


Running out of things

Hugh realizes he has a problem – he's maxed out the number of nodes. He talks to the local guru, and the guru says – no problem, just divide your CAN into two parts and join the parts with a CAN-to-CAN bridge. Hugh is doubtful, it can't be that simple, won't he need to change node ids and change his event programming? The guru reassures him – it is that simple – no need to change anything, the nodes will still see all the other nodes events, just as before. Hugh takes him at his word, picks up a bridge and, after five minutes deciding where to break the CAN bus, inserts the bridge and is running trains again.

One of the big problems with buses is that you eventually run out of things: id-numbers, bus capacity and bandwidth. Big events obviously takes care of the first, but S9.6 has also addresses the last two. If you have too many nodes or too much traffic, then you can break your layout into two bus-segments, and join them with a bridge-node, as diagrammed to the right.

This solves the problem of too many nodes on a segment, but not the traffic. S9.6 solves this by by running over fast transports, such as Ethernet. But this doesn't solve the problem if all the messages go everywhere. Partial solutions involve non-linear address spaces and limiting traffic on a message by message basis. S9.6 automatically limits traffic by only sending messages to those nodes that want them. This is called 'interested based filtering', and is illustrated to the right. The blue event is passed by the bridge-node, since two of the nodes on segment B are *interested* in the evemt. However, the red event has been blocked at the bridge, because no node requires it



on segment B. In addition, S9.6 messages are completely routable, and you can use cheap Ethernet routers to do the work. This is implemented now, and your layout can take up acres as space, and still run a single unified bus.

Other good works for Big layouts.

Hugh and John are in charge of setting up the modular meet. They use the configuration program to build a schematic diagram of the entire layout. They then drill down to each interface between layout segments and connect the edge nodes of each segment together. They quite quickly repeat the exercise on each adjacent pair of segments, with the help of the leads of the pairs of segments. Since every node has a unique ID and unique Events, they are not worried about conflicts or having to change ranges of events on any modules – reprogramming hundreds of nodes would not be inviting.

One of the major problems with large modular meets is the time taken to set up. S9.6 gives the meet organizers tools to shorten and ease this effort. Auto-configuration of modules and an automatic over-view of the layout is generated. The meet organizers can drill-down into each bus segment and get reports of all the nodes and their messages, so debugging and tweaking the layout is easy.