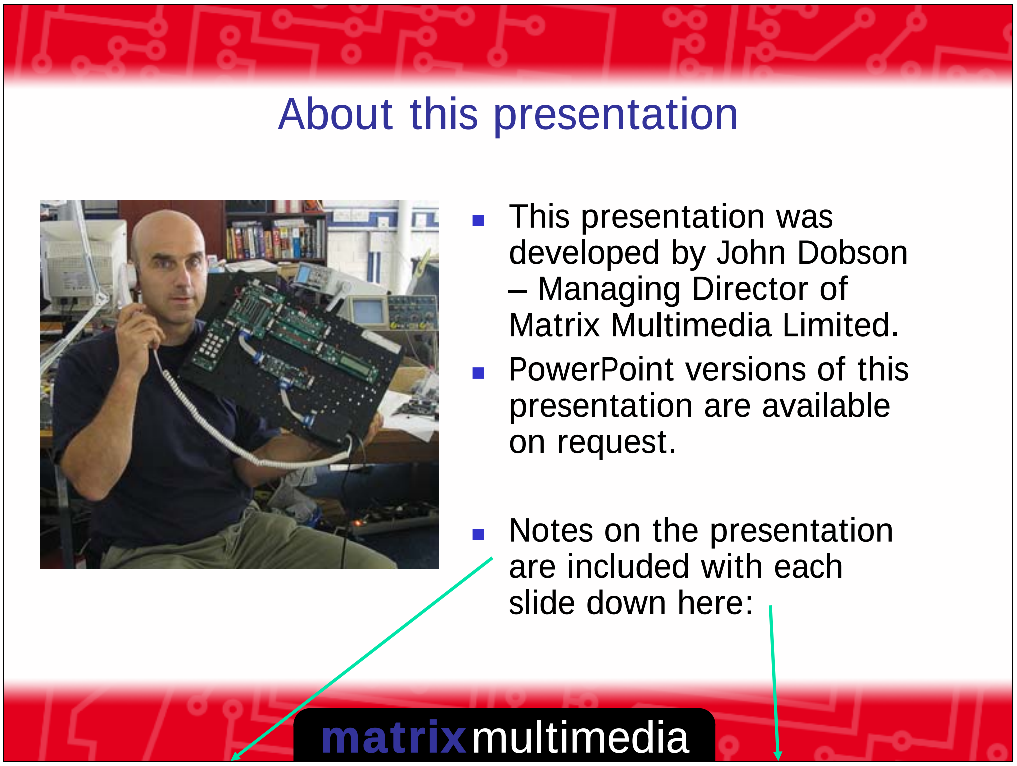
Automotive electronics – CAN and LIN buses

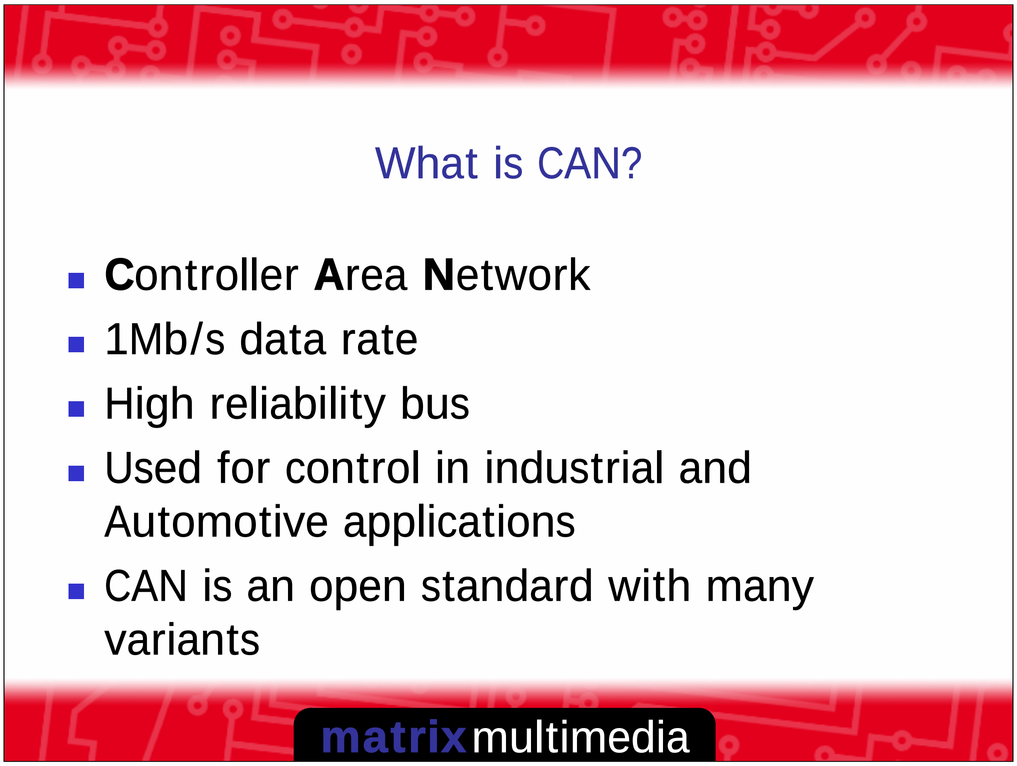
matrix multimedia



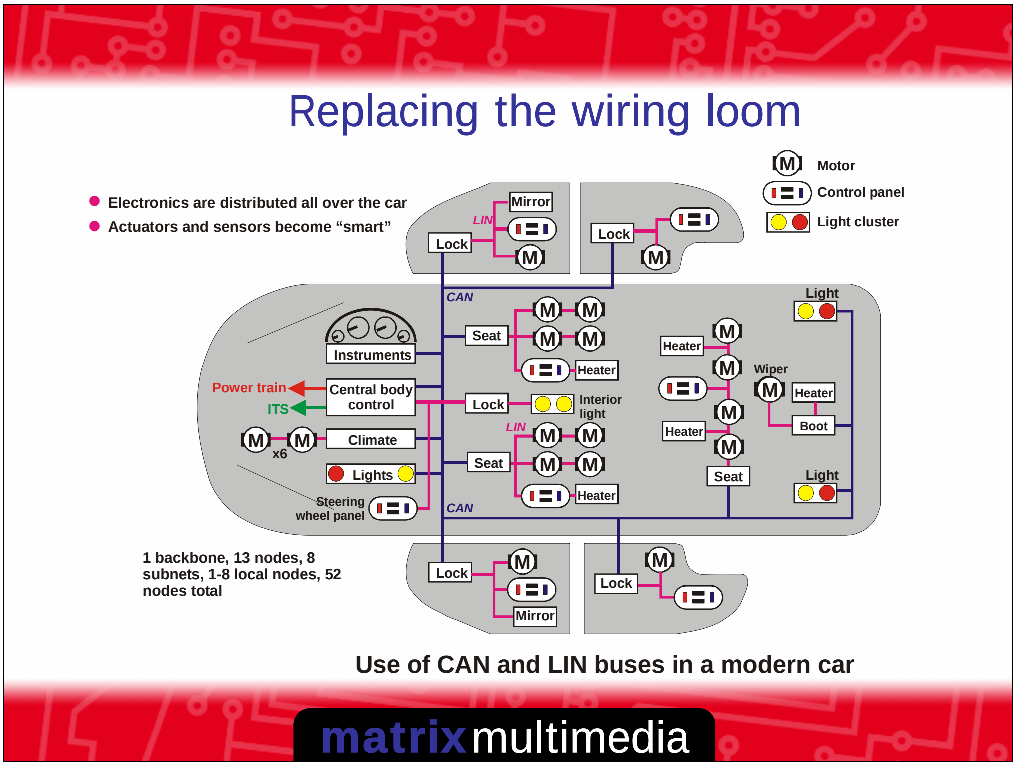
Automotive electronics is one of the few growth areas in electronics education. A major problem in this areas is understanding the control buses that have replaced wiring looms in vehicles. CAN bus is the most important of these – but many vehicles now use both CAN and LIN buses for cost reasons. We are developing solutions for both areas.

Part 1 - What is CAN?

matrix multimedia



CAN is also increasingly used in home automation and other areas. I have even had customers who are considering CAN for controlling points in small outdoor model railway systems.



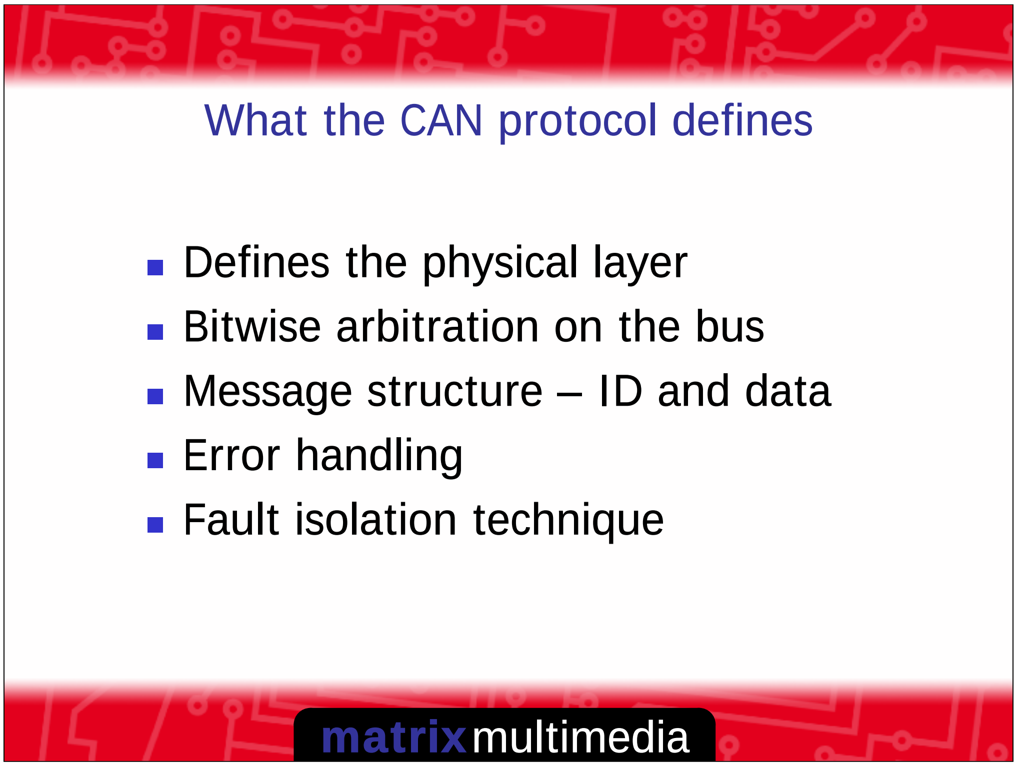
This diagram shows the distribution of CAN and LIN bus in a car. This is theoretical only – but based on current industry practice. LIN is slightly cheaper than CAN ($0.5 a node) and is lower data rate, lower reliability. Some automotive systems currently only use CAN – e.g. Massey Ferguson tractors. I am told that in some cars there are up to 300 microcontrollers. This seems unlikely at first, but if you consider that there is one in the tyre (for pressure) a pressure information receiver, an antilock brake microcontroller, then we are up to 12 already. Electrically operated seats probably have a few in etc. Maybe we can easily get to 150 micros, and maybe the 300 figure is true.

Advantages of CAN in automotive applications

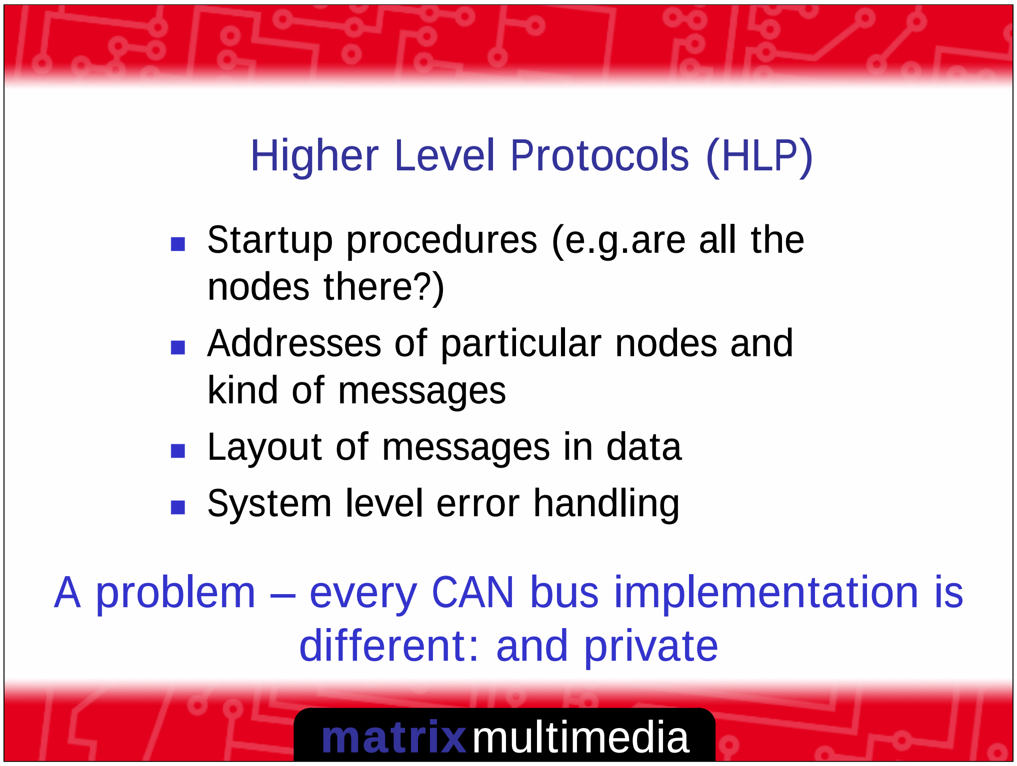
- Lower cost at vehicle construction
- Increased flexibility and reusability of design
- Reduces time to market
- Facilitates drive by wire which reduces cost further
- Facilitates advanced features in vehicles
- Enhances debug at point of service

matrix multimedia

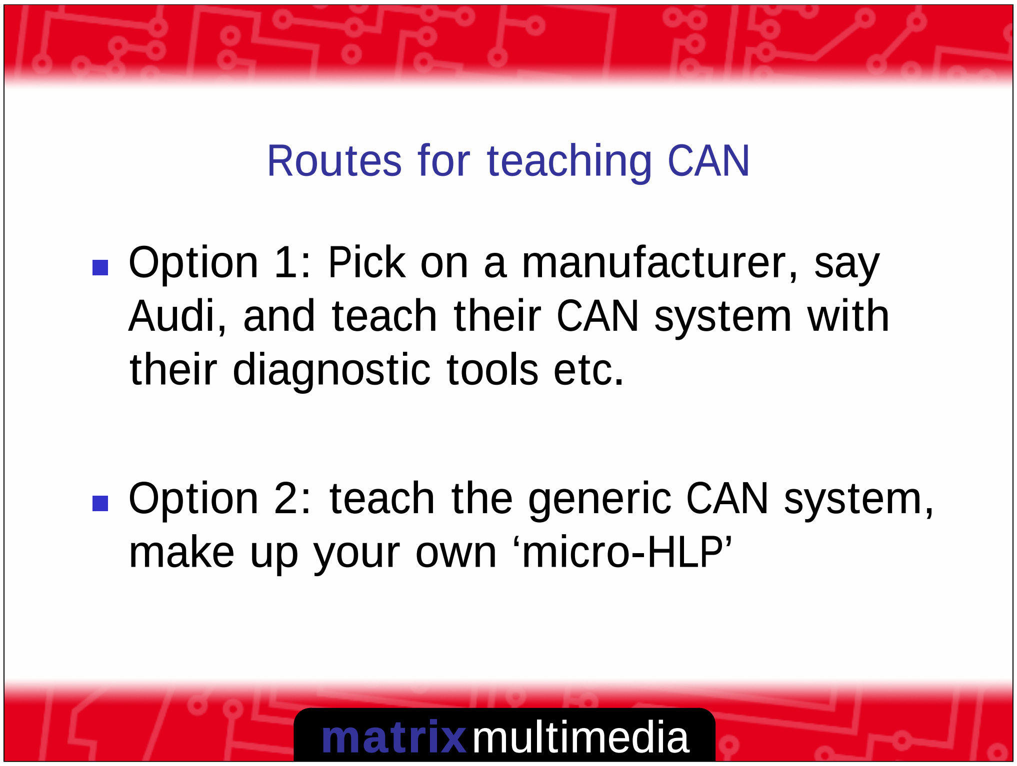
Cost and features are the big driving forces. (forgive the pun)



The low layers of the CAN communication protocol are defined as an international standard.



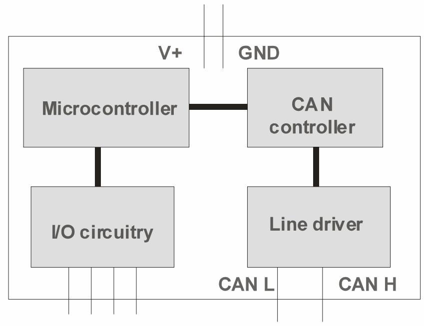
However each manufacturer will have their own upper layer definitions. A command for turning an indicator light on will be very different in a Ford car and an Audi car. Automotive manufactures may find it difficult to share HLPs and may not want to share them for commercial reasons. Also automotive manufacturers may not want to release details of the HLP for safety reasons – you don't want consumers messing with the technology that operates the antilock brakes, and the brake lights.



This is an areas of debate. Option 1 is incredibly expensive and very limited – students only experience one kind of CAN and may be sheltered by many of the details which help understand what CAN is and what it does. Option 1 has the advantage that the students would gain experience of using real diagnostic equipment, however they may be sheltered from details about how CAN actually works. In theory this is not a problem as automotive technicians only need to fix faults, however an understanding of the fundamentals is certainly an advantage. Our solution follows option 2.

A CAN ECU

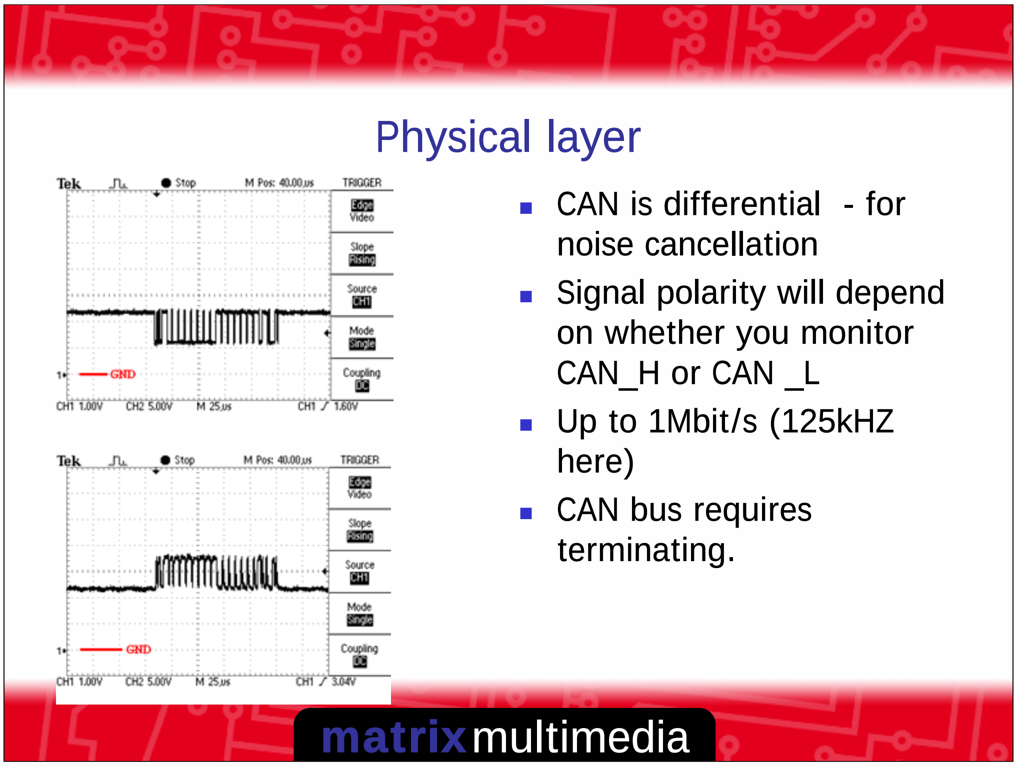
- A CAN ECU / node consists of a microcontroller, a CAN controller, a line driver and I/O
- This can be mimicked with E-blocks
- This can be controlled with Flowcode



E-blocks CAN controller and line driver

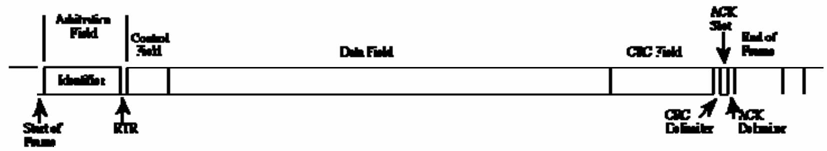
matrix multimedia

An Electronic Control Unit has the devices shown. In practice this can be implemented in a number of ways: for example you may find a microcontroller with an internal CAN controller and I/O circuitry and you may even find a single chip that has all these four blocks. The E-blocks solution is a bit blocky here, but they form a perfectly valid, fully working ECU.



Here you can see ‘scope traces of the signals on the two wires with respect to ground.

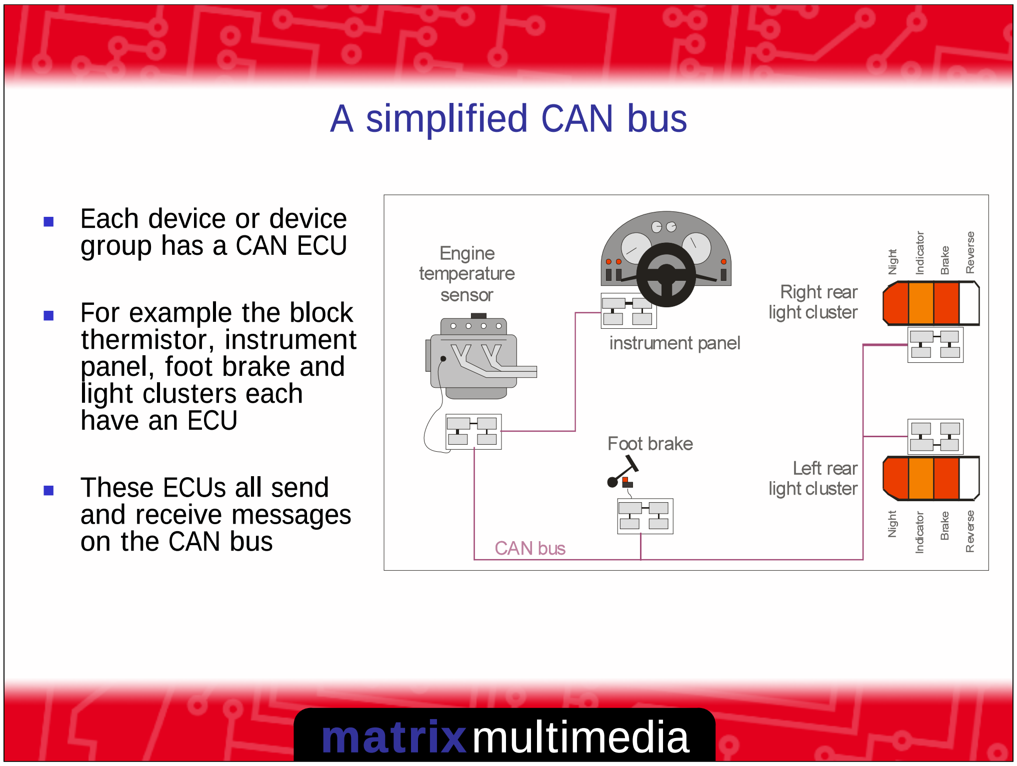
CAN packet structure



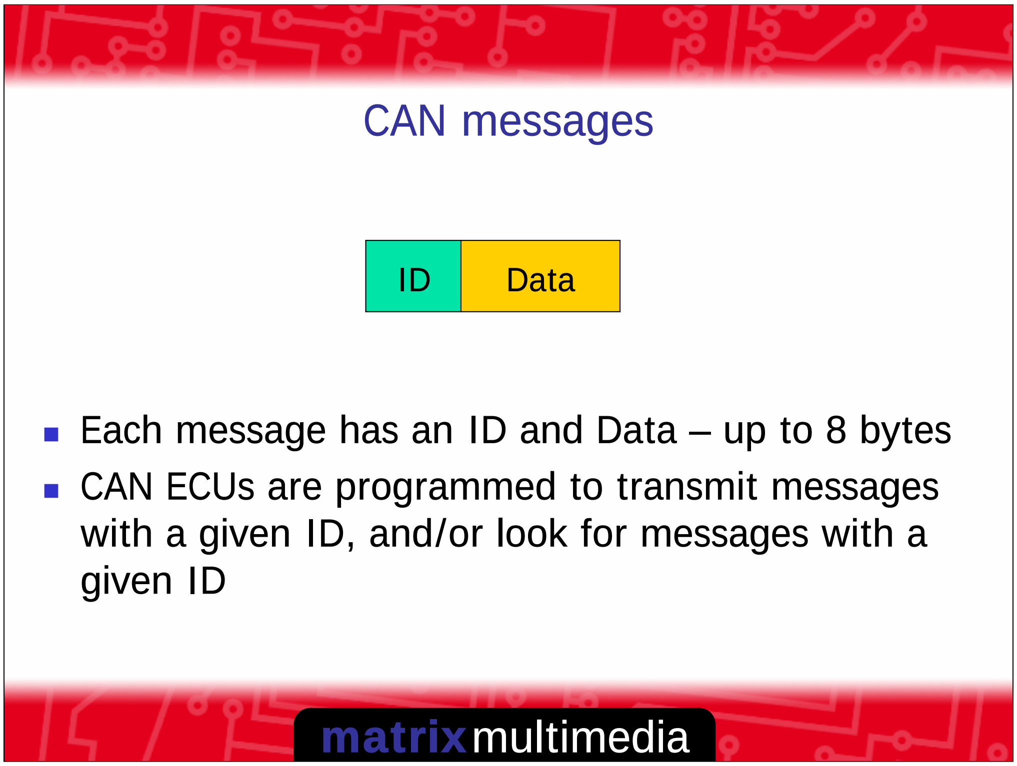
- Four types of messages:
  - Data frame, remote frame, error frame, overload frame
- We have implemented the Data frame, and remote frame: others are for bus management
- Data field length is adjustable
- A lot of this is taken care of by a CAN stack on the E-blocks board.

matrix multimedia

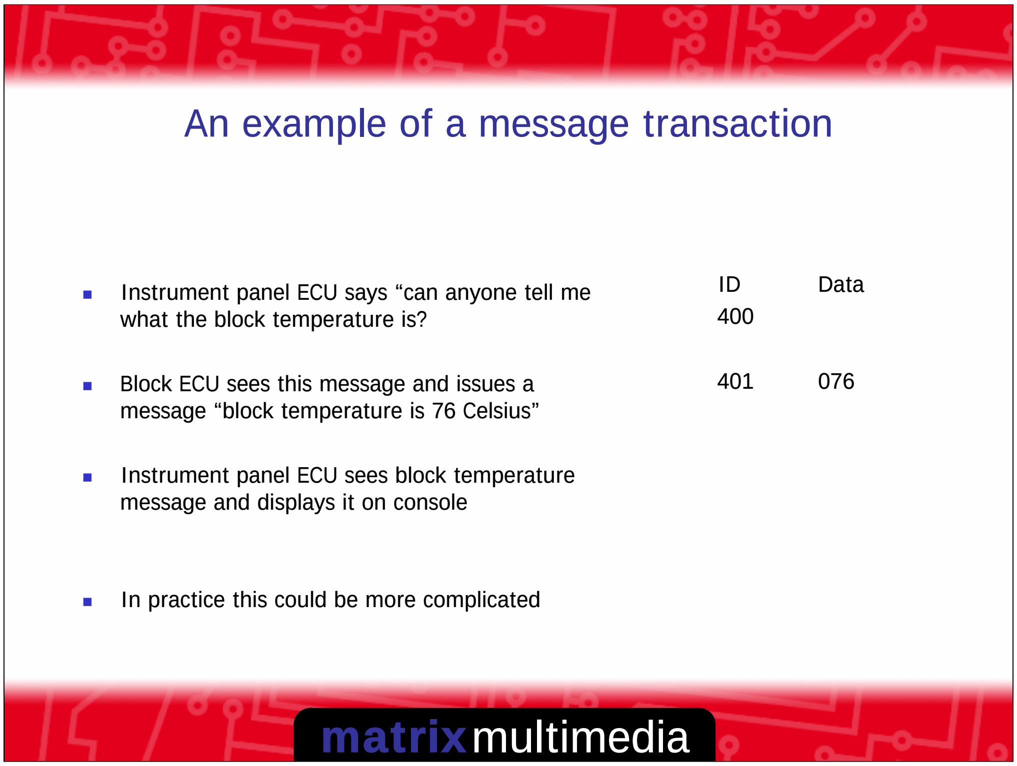
Here you can see the data frame for CAN. Several types are specified and we only support two.



The level of CAN ‘granularity’ will vary from manufacturer to manufacturer, and will also change with time as more devices become available. For example you may have one ECU in a light cluster, or you could have one ECU for each bulb within a cluster, or one ECU the light cluster and the rear wiper and wash unit.



This shows the simplicity of CAN. The bus just has messages with ID and data. Manufacturers then add functionality to the system by deciding what message to transmit on a particular event, and by deciding what actions should take place on receipt of a particular message. This customisation of the CAN bus system is referred to as a ‘higher level protocol’. Each manufacturer has its own higher level protocol.



This is an example of transactions on a CAN bus. The information is actually quite complex and specific – but it is distilled into very simple CAN bus transactions.

Things about CAN

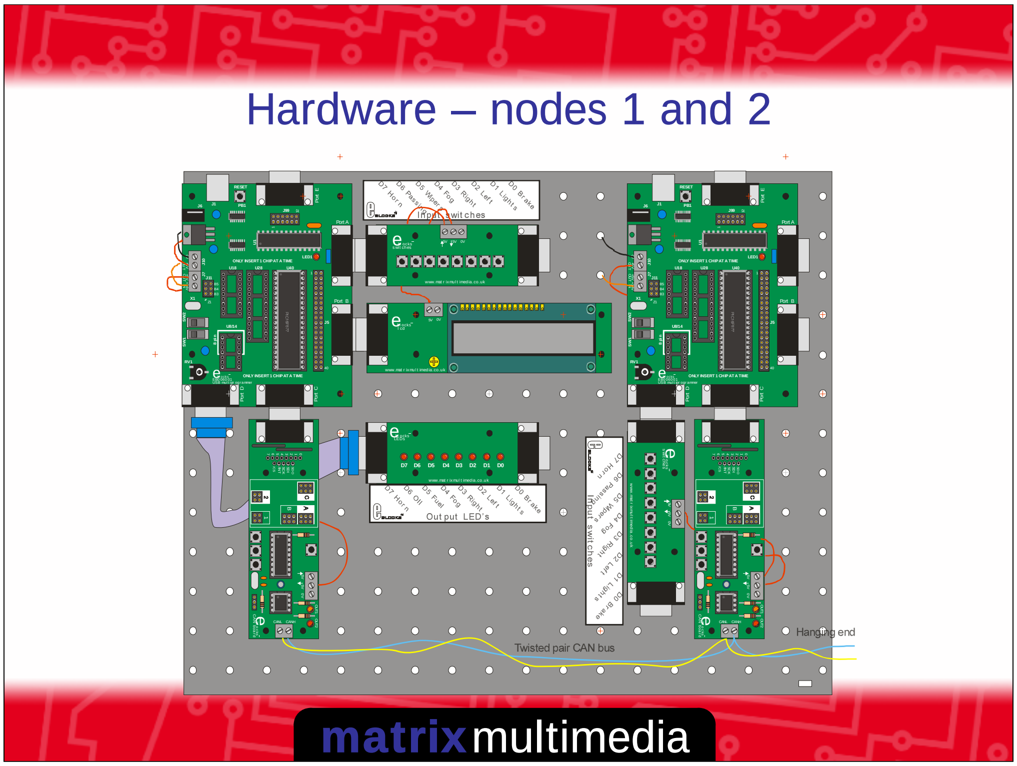
- The protocol is designed to increase integrity of the system
- There is no hierarchy
- It is designed for control – not transmission of large blocks of data
- At an basic level the protocol is quite simple – the details are quite complex

matrix multimedia

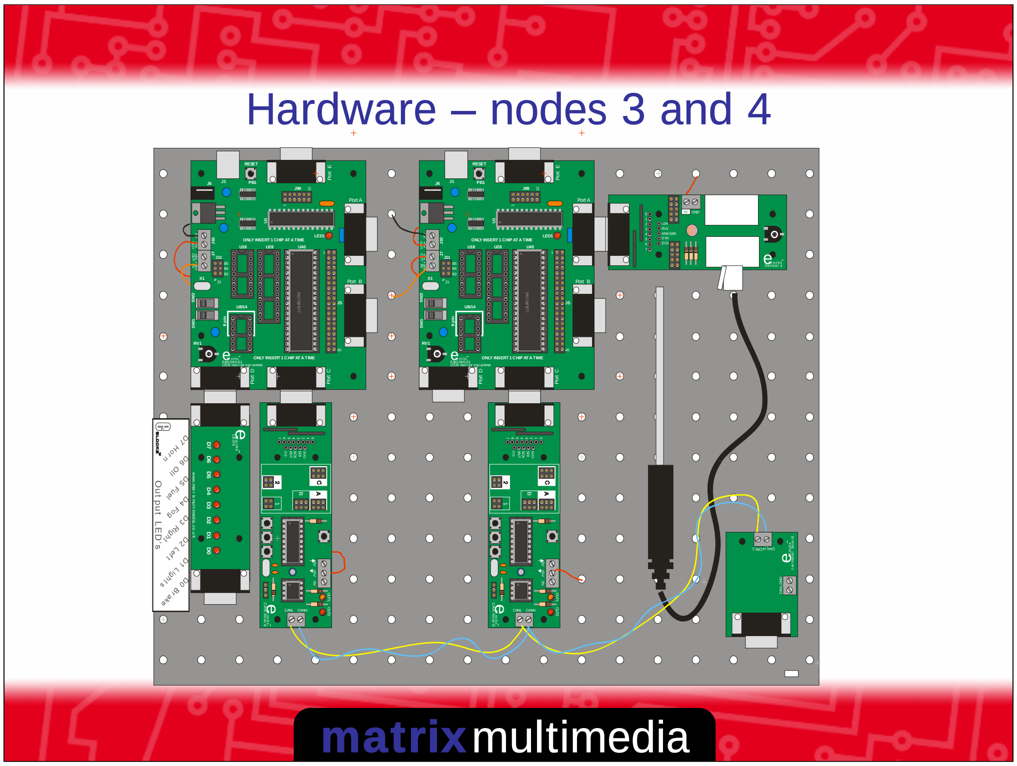
From the previous transaction you can see that these are key features of CAN.

Part 2 – the E-blocks CAN bus solution

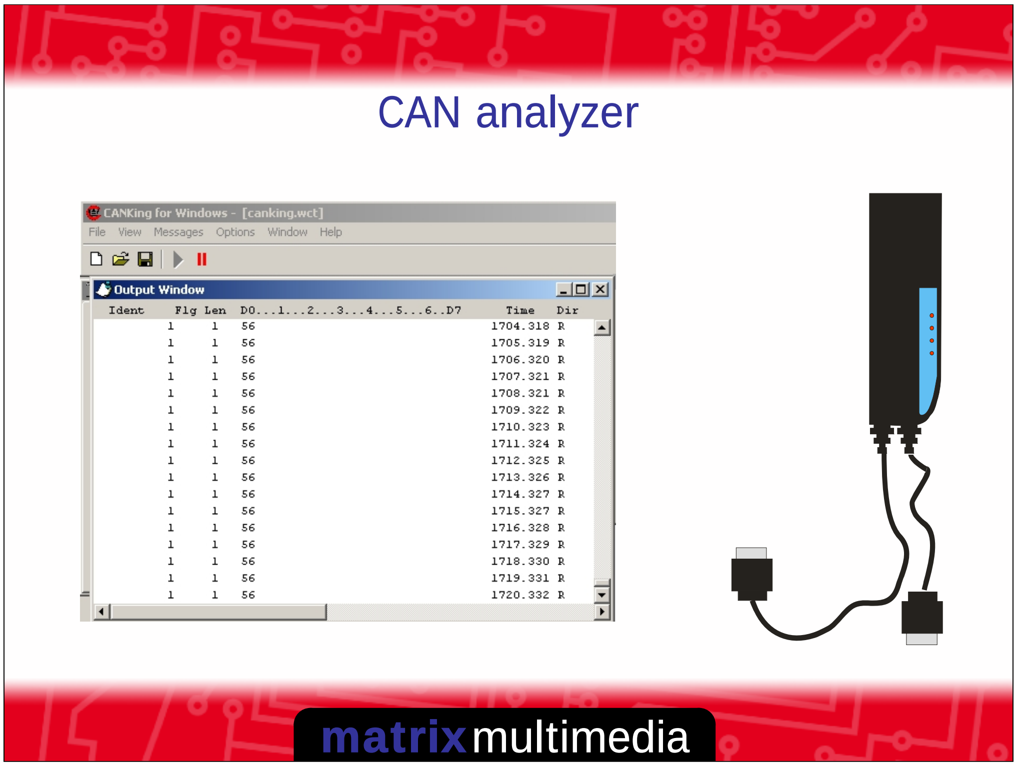
matrix multimedia



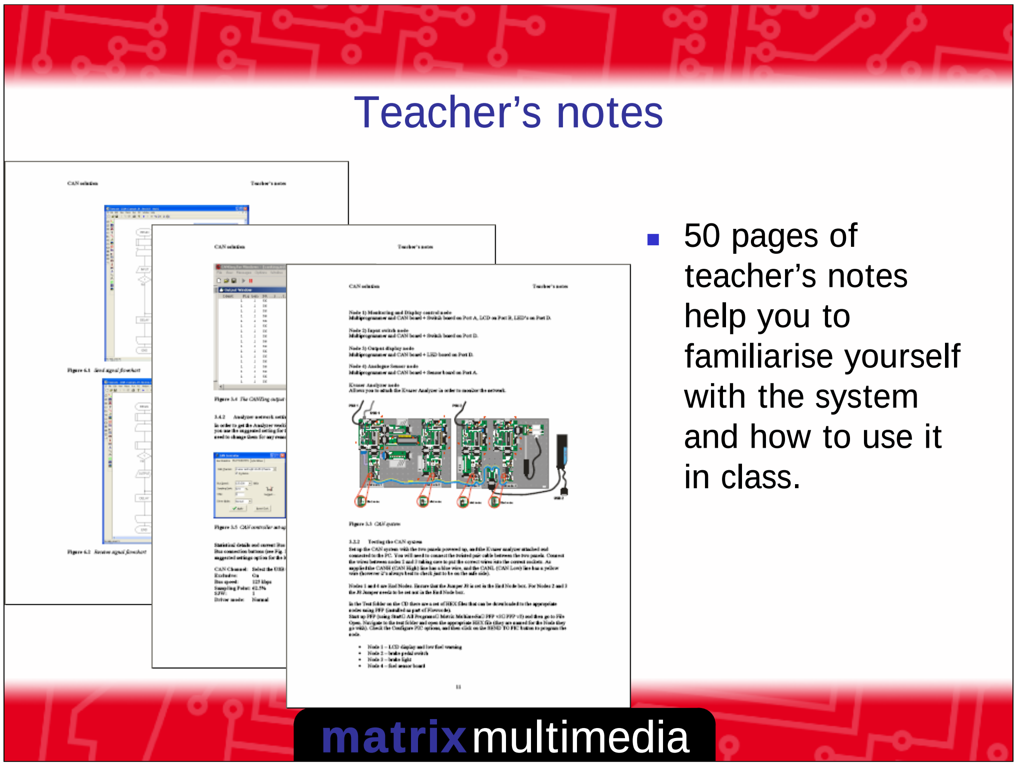
Node 1 is mimics the dashboard. Node 2 is a general purpose switch panel for brakes, indicator control etc.



Node 3 is a general purpose LED node to mimic rear light clusters etc. Node 4 is a sensor node where you can mimic an ECU that measures block temperature. The small circuit board allows the CAN analyzer to be plugged in.



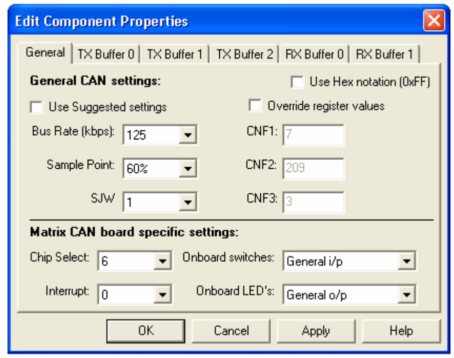
The Can analyzer generates CAN messages and also shows the CAN messages on the bus. Here you can see a simple screen shat shows messages with ID 1, one data byte of value 56.



As part of the solution we also ship a manual that covers setting up the system and gives some sample exercises for students.

Flowcode CAN software

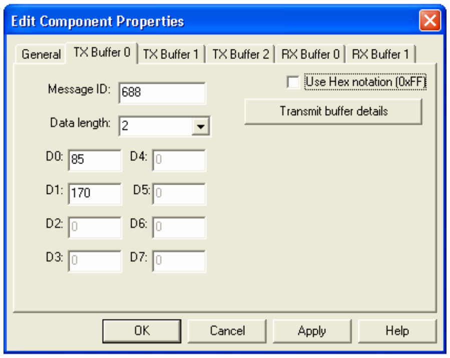
- High level software in Flowcode
- Caters for different levels of user
- Makes CAN transactions easy
- Can be used by automotive technicians upwards



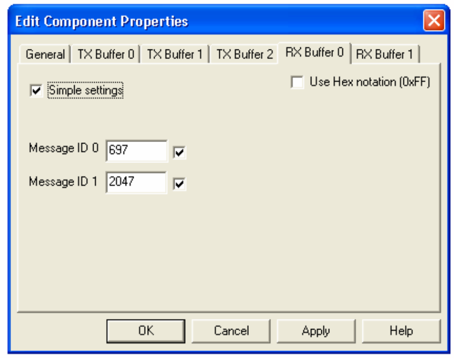
General settings screen

Flowcode CAN software is a key part of this. The software provides access to the CAN hardware at a high level so that students can concentrate on setting up the CAN system and not spend all their time on low level programming.

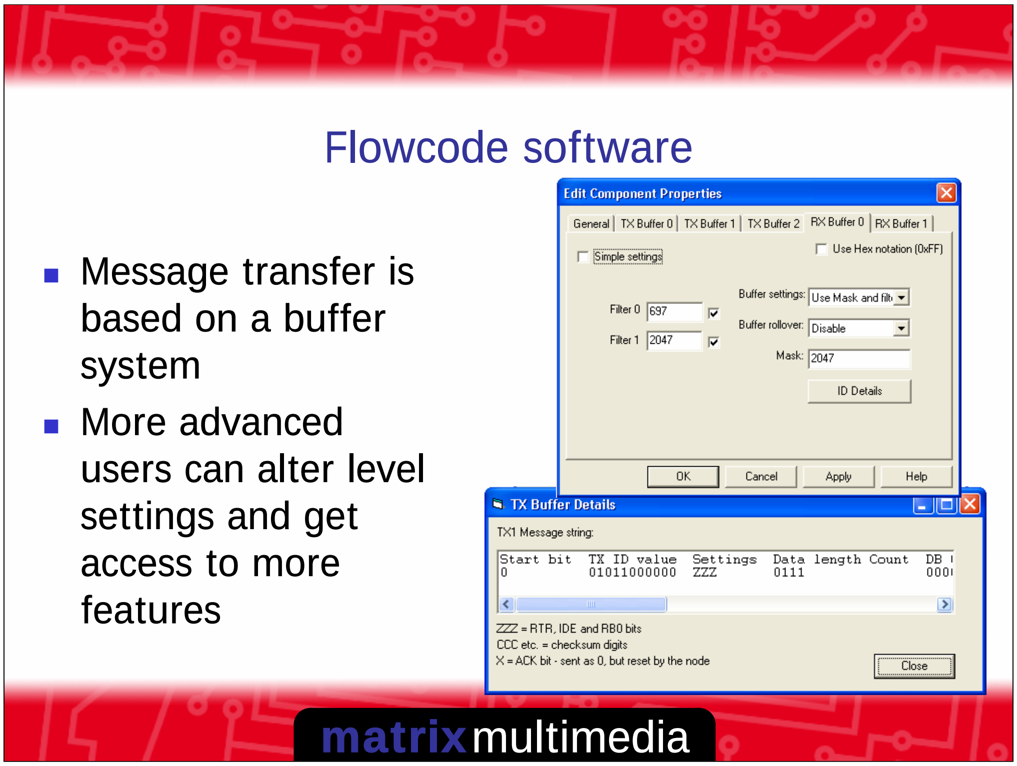
Flowcode CAN software



Transmit settings screen



Receive settings screen



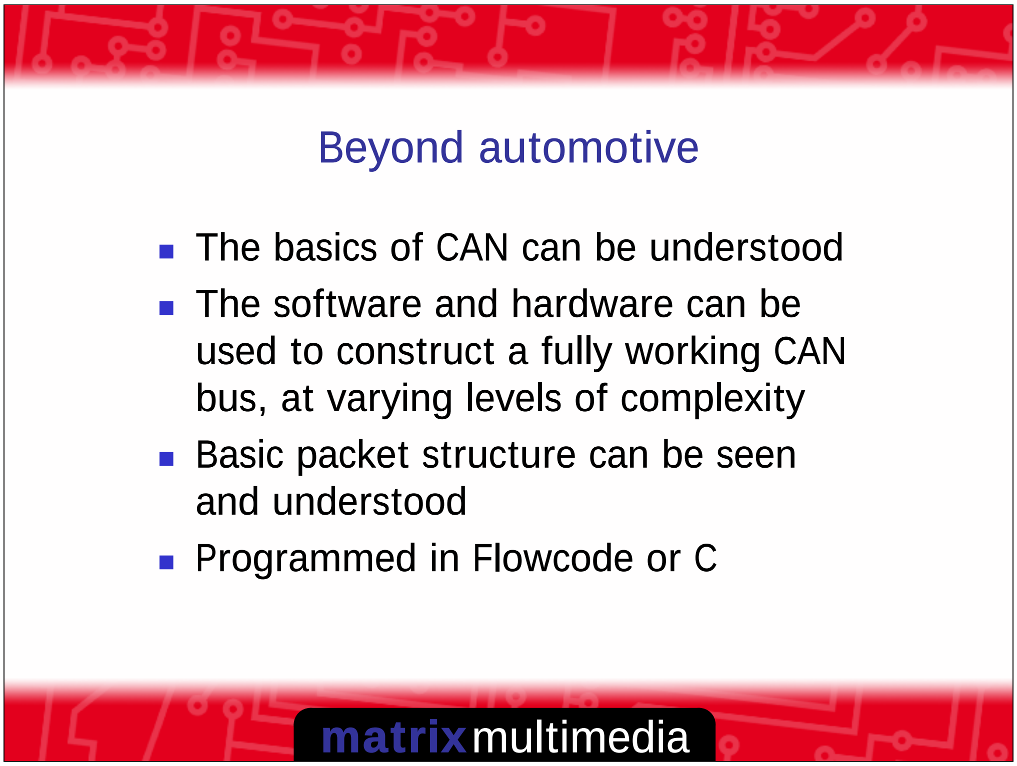
These additional screens show how the CAN system operates with buffers – transmit and receive. When a message is received it goes into a buffer very quickly. The microcontroller has to poll the CAN controller device regularly to see if there are any relevant messages. It is possible to set the buffers to filter out messages with a given ID. The contents of the transmit buffer can be examined so that students can understand packet structure.

CAN for Automotive students

- Students can understand the nature of an ECU
- Students can understand that ECUs can be reprogrammed and have hardware and software faults
- Students can construct basic CAN systems – with prewritten programmes
- Students can construct simple CAN systems

matrix multimedia

Theses are some of the requirements of automotive technicians. The CAN solution is ideal for this.



Advanced users can use C to control all aspects of the CAN bus. However the code behind this could be ambitious! IN practice we believe that even degree level students don’t need to understand much more than the basics of CAN – asking degree students to program CAN systems in C may be more of a final year project than a learning exercise.

At this point in the presentation I make a demonstration of how you can sue Flowcode to construct CAN systems and how we can use The packet analyser to see the traffic on the CAN network.

Part 4 – LIN bus

Node 1 – ID100, data 85.

matrix multimedia

LIN status report

- We are just starting out with LIN
- We know what we want and how to do it – we are just not finished yet
- Here is what we currently have:

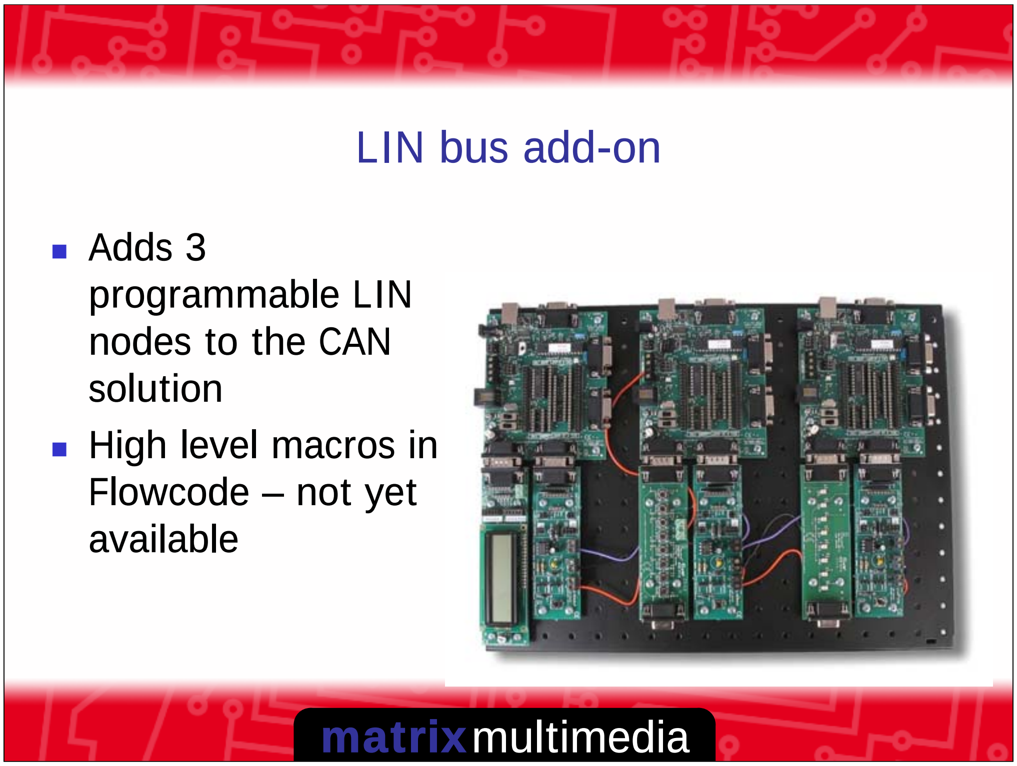
matrix multimedia

LIN bus

- 3 wire bus – GND. +12V and LIN
- Low data rate, low reliability
- Saves $0.5 per node on CAN
- Additional panel on the CAN solution will cover LIN

matrixmultimedia

We are in the process of developing a LIN bus board for E-blocks. This will work in much the same way as the CAN board with high level macros for Flowcode.



The LIN bus connects to the CAN bus (node 4 on the CAN bus becomes a CAN/LIN bus node) to provide 3 LIN nodes. Flowcode macros are under development that allow students to gain experience of LIN.