USB3_2 An interface module between CBUS and USB.

Introduction

USB3_2 is one of a number of modules for use with the CBUS system. This is a general purpose layout control bus (LCB) using the industry standard CAN bus. For more information on CBUS, see the introductory article on the website (www.cbus.org.uk).

USB3_2 is unique amongst the CBUS modules in that it is neither a producer nor consumer of events but a simple interface between CBUS and a standard USB port. Its primary function is to link CBUS with a computer (PC) or other device using a USB serial protocol. It acts as a two way message passing node.

This is the third hardware design for the CAN_USB module. CAN_USB2 uses the FTDI FT245RL integrated circuit which is a surface mount device. This is suited to constructors able to solder fine pitch SM circuits. The alternative CAN_USB3 uses a ready constructed USB interface module (FTDI UM245R) which plugs into a conventional IC socket. This is a reasonably priced module and avoids the need for SM soldering and sourcing of components like the USB 'B' socket. The USB3_2 is effectively a CAN_USB3 with on on-board 5V power supply.

Please refer to the schematic USB3_2_sch.pdf

Power supply.

This module requires a 12V DC supply. Note that all other _2 CBUS modules require a similar 12V DC supply and the same supply can be used for all.

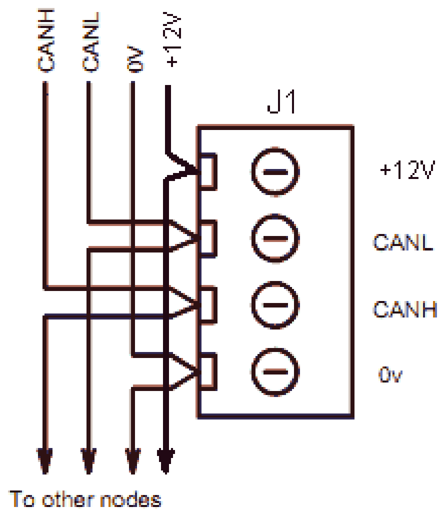
The USB3_2 module cannot be powered from the USB cable. Unfortunately, some USB connections employ a 'suspend' mechanism which disconnects power if there has been no activity for a pre-set time. Normally, the USB module is reactivated by data from the PC. When used for monitoring traffic on the CBUS, the data is from the USB module to the PC so it does not 'wake up'. On the UM245R module, leave the jumper J1 between pins 2 and 3 but remove jumper J2. For more information, download the data sheet from

http://www.ftdichip.com/Products/EvaluationKits/UM245R.htm

The green LED (LD2) will illuminate to show the circuit is working correctly. This is not just a power on indicator but confirms correct working of the processor.

Connecting the module

The CANH and CANL wires go to all modules. They are polarity sensitive so CANH must go to CANH and CANL to CANL. These wires should preferably be a twisted pair but it is not essential, especially for short distances.
The 0V must also be connected to all modules.



Where a number of modules are powered off the same 12 V DC supply, then their 12V input terminals should all be connected together.

While it would be usual to wire the bus sequentially round the various modules, it is not essential and individual nodes can be 'star' connected if this is more convenient.

The CAN bus requires 'termination' resistors at some point in the network. If the bus is wired sequentially round the modules, then a resistor of 120 ohms should be fitted across the bus at each end. For small layouts, it is sufficient to have a resistor across the bus at one point. The value is not critical and a 68 ohm resistor will suffice.

Note: Where the CBUS contains a mixture of the older AC powered modules and the current DC powered modules the 12V DC should be independent of the AC supply, **not** derived from the same transformer. Only the CANH, CANL and 0V should be connected between AC and DC powered modules.

The USB interface.

The connection is via a standard A to B USB cable. Provided the USB driver has been installed, the PC will automatically detect the USB3_2 module when it is connected. However, the USB3_2 module must be powered before the PC will recognise it. It also needs to be powered while installing the FTDI CDM drivers.

http://www.ftdichip.com/Drivers/VCP.htm

and click on 2.06.02 for the latest version.

See the document 'CAN_USB2.pdf' for driver installation instructions.

The serial protocol.

There are several published CAN to serial protocols but we adopted the one by 'Gridconnect' to whom full acknowledgement is made. The full protocol is available from

http://site.gridconnect.com/docs/CAN/can-rs232.pdf

although we have only used the basic message format. This is described below.

The information on the serial side uses ASCII characters. This simplifies message parsing by the PC and is compatible with most software, e.g. Visual Basic. However, the structure of the ASCII string follows that of a CAN frame so there is direct correspondence between the CAN frame and the serial string.

The header.

Following the 'Gridconnect' scheme, the ASCII string starts with a ":" followed by an "S" to indicate a Standard CAN frame or an "X" for an extended frame. CBUS only uses Standard frames but the USB3_2 module allows for both types of frame. The next 4 chars are the ASCII version of the two header bytes in HEX for a standard frame or 8 chars for the four bytes of an extended header. This is departure from the Gridconnect format as CBUS uses a 7 bit node ID and 4 priority bits rather than just an 11 bit number. These two bytes map directly into the bytes sent and received by the CAN processor as SIDH and SIDL. (Standard IDentifier High byte and Standard IDentifier Low byte) For an extended header, the four bytes map directly to the SIDH, SIDL, EIDH and EIDL.

An example would be where the CBUS priority bits are 1011 and the CAN ID number is 0000001. These bits become the two bytes of the CAN header as follows
10110000 00100000 or in HEX form, B020. SIDH is B0 and SIDL is 20. This gives the string so far as :SB020 or in ASCII,

3A 53 42 30 32 30

The frame type

The next character is either "N" or "R" signifying a Normal or a RTR frame (RTR is Remote Transfer Request). Except during the self enumeration process, CBUS only uses Normal frames.

The data segment

A CBUS frame has up to 8 data bytes and the remainder of the string is the data bytes in ASCII (HEX) form. The string is concluded by a ";" Note, there is no value indicating the number of data bytes. This is worked out by the firmware in the USB3_2 module. If a frame has all 8 data bytes then the format for a normal frame is as follows.

:ShhhhNd0d1d2d3d4d5d6d7;

Where hhhh is the two byte header and d0 to d7 are the 8 data bytes. If the header is B020 as above and the data is 1,2,3,4,5,6,7,8 then the ASCII string becomes

3A 53 42 30 32 30 4E 30 31 30 32 30 33 30 34 30 35 30 36 30 37 30 38 3B

Exactly the same format is used for data to or from the CAN-USB module.

A PC software program is available which allows entry of the various bytes in either HEX or binary and then sends the appropriate ASCII string. The same software also displays incoming CBUS frames in HEX and binary and also whether the frame is N or R and the number of data bytes. This program is written in Visual Basic 5 but a compiled (installable) version is also available. See the MERG website. The USB3_2 module can also be used with the JMRI (JAVA Model Railroad Interface) package.

Notes.

The USB3_2 module, when used with suitable PC software, is fast enough to stream CAN frames at the full rate. It is able to act as a true ‘sniffer’ without missing any CAN frames.

The pushbutton S1 on the PCB is not presently used on the USB3_2 module. The USB3_2 firmware has the CAN_ID set by the PC with the appropriate bits in the header section. This allows for multiple USB3_2 modules with multiple PCs. The PC program should ensure that the CAN_ID is different for each USB3_2 module as well as different from any other ‘producer’ module.

The priority bits are also sent so the PC can determine the priority of the outgoing message. (The CAN ID is not part of the CBUS message). However, for received CBUS messages, the CAN ID sent to the PC is that of the received frame. This allows for monitoring other node CAN IDs as well as their message priority.

The PCBs include provision for in-circuit serial programming and debugging (ICSP). LD2 (green) should be steady green in normal operation.

The alteration with rev g and subsequent revisions of the firmware to allow for both standard and extended frames makes this module suitable for any CAN system. It can also be used with the CBUS bootloader which uses extended frames.

Resistor R3 determines the rise and fall times of the CAN waveform. The value should be 100K to minimise fast edges and possible EMI.

The full schematic, a PCB layout which is in .PDF form and can be printed to the exact size for making masks and the PIC assembly and HEX code are available on the MERG website.
These can be freely used for non-commercial purposes. Copyright to the designs is held by the authors.