LED_2 CBUS module for driving 64 LEDs.

Introduction

LED_2 is one of a number of layout control 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)

LED_2 is a ‘consumer’ only module which accepts and learns events sent over the CBUS by other (producer) modules. It drives 64 separate LEDs using a 4 row by 16 column matrix. The column driver IC has a resistor to enable the LED current (brightness) to be adjusted. The column driver is a current sink so no external LED resistors are required.

LED_2 can learn and store 254 separate events and any event can turn on or off an individual LED. Additionally, the polarity of the response relative to the command can be set for each event so an ON event can turn a LED off and v.v. An extra feature of the CANLED module is a ‘toggle’ mode where an ON event to one LED will turn off the next LED in sequence and similarly the other way round for an OFF event. This can be set for any LED pair and doesn’t apply to all the LEDs at once so some can work as toggled pairs and some as individual LEDs. This toggle facility is particularly intended for control panel use where a switch sending ON and OFF commands for a single event will operate two LEDs to indicate the switch or turnout position.

LED_2 defaults to the SLiM (Small Layout interface Model) of CBUS which allows it to be set up and taught without any need for a programming device or computer. However, like all other SLiM modules, it responds with its CAN-ID when interrogated by ‘nodes’ which implement the self-enumeration scheme so is compatible with modules running in the Full Model (FLiM). Firmware version **LED2_k** and above allows for running in either mode.

Please refer to the schematic LED_2_sch.pdf.

Power supply

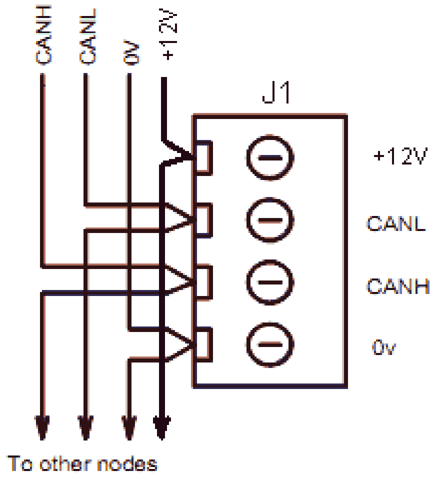
LED_2 in common with all _2 modules requires an external 12V DC supply. A common 12V supply can be used for all _2 CBUS modules. Power input is via the CBUS connector pins 1 and 4.

There is a separate voltage regulator (U2) which supplies the rest of the circuit with a fixed 5V. The maximum current available from this regulator is 1 Amp. With all 64 LEDs illuminated and an average current of 10mA per LED, the module current is about 700mA. The power dissipated in the 5V regulator is 5 watts with this load and even with only 50% utilisation is 2.5 watts. A heatsink of not more than 100 C / watt is recommended.

To reduce the heating, it is recommended that low current LEDs be used. Increasing R2 to 1K8 will set the LED current to about 3mA.

In SLiM mode, 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. When in FLiM mode the yellow LED (LD1) shows correct running.

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. Screening is not necessary. The 0V must also be connected to all modules.

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

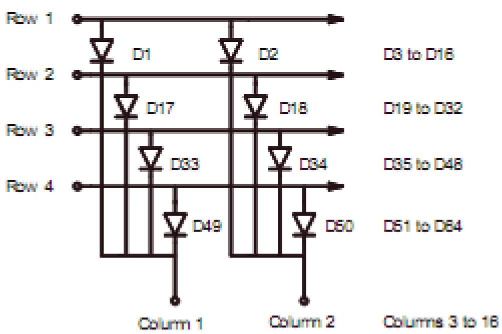
While it would be usual to wire the bus sequentially round the various modules, it is not essential and individual nodes can be 'star' connected if this is more convenient. The CBUS wiring should be kept separated from any DCC supply wiring to prevent possible interference.

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.

Connecting the LEDs.

The LEDs use a multiplexed array. This has 4 rows and 16 columns.

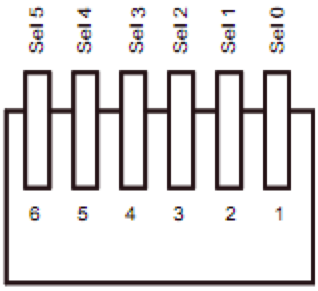


LED matrix connections. (two columns shown)

The LED wiring connects to the D type socket on the PCB. Only 20 wires are required for all 64 LEDs. Refer to the schematic for the relationship between the row/column numbers and the socket connections. The average LED current is set by resistor R2. The suggested value gives an effective current of 10mA. If low current LEDs are used, then the average current may be reduced by increasing this resistor. This will also affect the brightness.

Training the module in SLiM mode.

The training process is a unique aspect of CBUS and provides a very powerful method of configuration without the need for any programming device or knowledge of how the system works. The LED_2 module has a 6 way DIL switch for selecting the LED and a group of four ‘jumpers’ for other functions.



DIL switch Top view

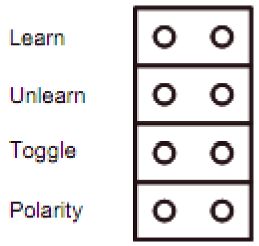
The 6 ‘Sel’ switches select which LED the ‘event’ will apply to. There are 64 LEDs numbered 1 to 64. The six switches allow a selection of one of the 64 LEDs using a binary sequence. When the switch is ‘down’ (ON as written on the switch) this represents a logic 0. A switch in the up (OFF) position is a logic 1. With all six switches down, this gives a value of binary 000000 and selects LED 1. All switches up gives 111111 and selects LED 64

The following table gives all the possible combinations.

Sel 5	Sel 4	Sel 3	Sel 2	Sel 1	Sel 0	LED		Sel 5	Sel 4	Sel 3	Sel 2	Sel 1	Sel 0	LED
on	on	on	on	on	on	1		off	on	on	on	on	on	33
on	on	on	on	on	off	2		off	on	on	on	on	off	34
on	on	on	on	off	on	3		off	on	on	on	off	on	35
on	on	on	on	off	off	4		off	on	on	on	off	off	36
on	on	on	off	on	on	5		off	on	on	off	on	on	37
on	on	on	off	on	off	6		off	on	on	off	on	off	38
on	on	on	off	off	on	7		off	on	on	off	off	on	39
on	on	on	off	off	off	8		off	on	on	off	off	off	40
on	on	off	on	on	on	9		off	on	off	on	on	on	41
on	on	off	on	on	off	10		off	on	off	on	on	off	42
on	on	off	on	off	on	11		off	on	off	on	off	on	43
on	on	off	on	off	off	12		off	on	off	on	off	off	44
on	on	off	off	on	on	13		off	on	off	off	on	on	45
on	on	off	off	on	off	14		off	on	off	off	on	off	46
on	on	off	off	off	on	15		off	on	off	off	off	on	47
on	on	off	off	off	off	16		off	on	off	off	off	off	48
on	off	on	on	on	on	17		off	off	on	on	on	on	49
on	off	on	on	on	off	18		off	off	on	on	on	off	50
on	off	on	on	off	on	19		off	off	on	on	off	on	51
on	off	on	on	off	off	20		off	off	on	on	off	off	52
on	off	on	off	on	on	21		off	off	on	off	on	on	53
on	off	on	off	on	off	22		off	off	on	off	on	off	54
on	off	on	off	off	on	23		off	off	on	off	off	on	55
on	off	on	off	off	off	24		off	off	on	off	off	off	56
on	off	off	on	on	on	25		off	off	off	on	on	on	57
on	off	off	on	on	off	26		off	off	off	on	on	off	58
on	off	off	on	off	on	27		off	off	off	on	off	on	59
on	off	off	on	off	off	28		off	off	off	on	off	off	60
on	off	off	off	on	on	29		off	off	off	off	on	on	61
on	off	off	off	on	off	30		off	off	off	off	on	off	62
on	off	off	off	off	on	31		off	off	off	off	off	on	63

on	off	off	off	off	off	32		off	off	off	off	off	off	64

There is also a 4 position jumper block on the PCB. This has 4 functions. Learn, unlearn, polarity and toggle. A jumper in gives a logic low or an ON state.



The jumper block

To train the CANLED module, you need a CBUS ‘producer’ module which creates events. This could be an ACE3_2 (CANACE3) control panel scanner, an ACE8C_2 (CANACE8C) switch input module or a PC program which creates events via a CAN-RS/USB module.

Connect the various modules and apply power. Connect the LED array. Select the LED you want the event to operate with the Sel switches according to the above table. Put the ‘learn’ jumper in (ON). Send the event. If it is an ON event, then the corresponding LED will turn on. Try it with the same event number but as an OFF event. The LED should now go off. Remove the learn jumper. Test the event again. The LED should go on or off as expected.

The next option is the use of the ‘Pol’ jumper. With the Pol jumper out, an ON event will turn a LED on and vice versa. However, it is possible to reverse this by putting the Pol jumper in when learning the event. Now an OFF event should turn the LED on and an ON event turn it off.

The other option with the LED_2 module is the ability to have a single event toggle two adjacent LEDs. This is particularly intended for control panels where a switch or pushbutton sets a turnout using a single ON or OFF event to set the direction. This same event can now indicate the turnout position using two LEDs. With both the learn and toggle jumpers in, set the LED switches to the first LED of the pair. Send the ON event. The first LED will light. Now send the same event as an OFF. The next LED in sequence will light and the first one will go off. If the polarity jumper is also fitted, the sequence is reversed.

Note that it is possible to set another event which will actuate either of the same LEDs separately. When using the toggle facility, it is best to consider the LEDs as pairs and number them alternately. However, not all the LEDs may be required to work in pairs, e.g. when setting a route so this option remains.

If you want the module to forget an event it has learned, (remove it altogether), set both the learn and unlearn switches ON and send that event.

Note that different events can set the same LED. This can be useful if you want switches on different control panels or a combination of control panel and PC events to have the same effect.

Notes.

The present firmware sets the number of stored events to 254. If you try to set more than this, nothing will happen. This allows nominally 4 events per LED but you can have more for some and less for others up to the 254 maximum.

You can alter the polarity for an existing LED or put it in toggle mode but you cannot remove a LED from an event once set. If you want to remove a LED, you need to erase the whole event and teach it again.

If you have forgotten which events the module has been taught, then you cannot tell it to unlearn an event that you don't know it has! In this case there is a 'clear all' option. This removes completely all stored events. To do this, set the unlearn jumper in with the learn jumper out . Remove power from the module and then reapply it. This will clear the memory. Remember to remove the unlearn jumper after this process.

The PCBs include provision for in-circuit serial programming and debugging (ICSP). The firmware also contains code for use with the CBUS bootloader so code can be updated over the CBUS. (see document 'CBUS_Bootloader.pdf)

The small pushbutton S1 is for switching between the SLiM mode and the Full (FLiM) mode. For this functionality firmware version **LED2_k** or higher is required.

Resistor R7 determines the rise and fall times of the CAN waveform. The value should be 100K to minimise fast edges and possible EMI.

For operation in the Full mode (FLiM), see the document 'Full_mode_operation.pdf

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.