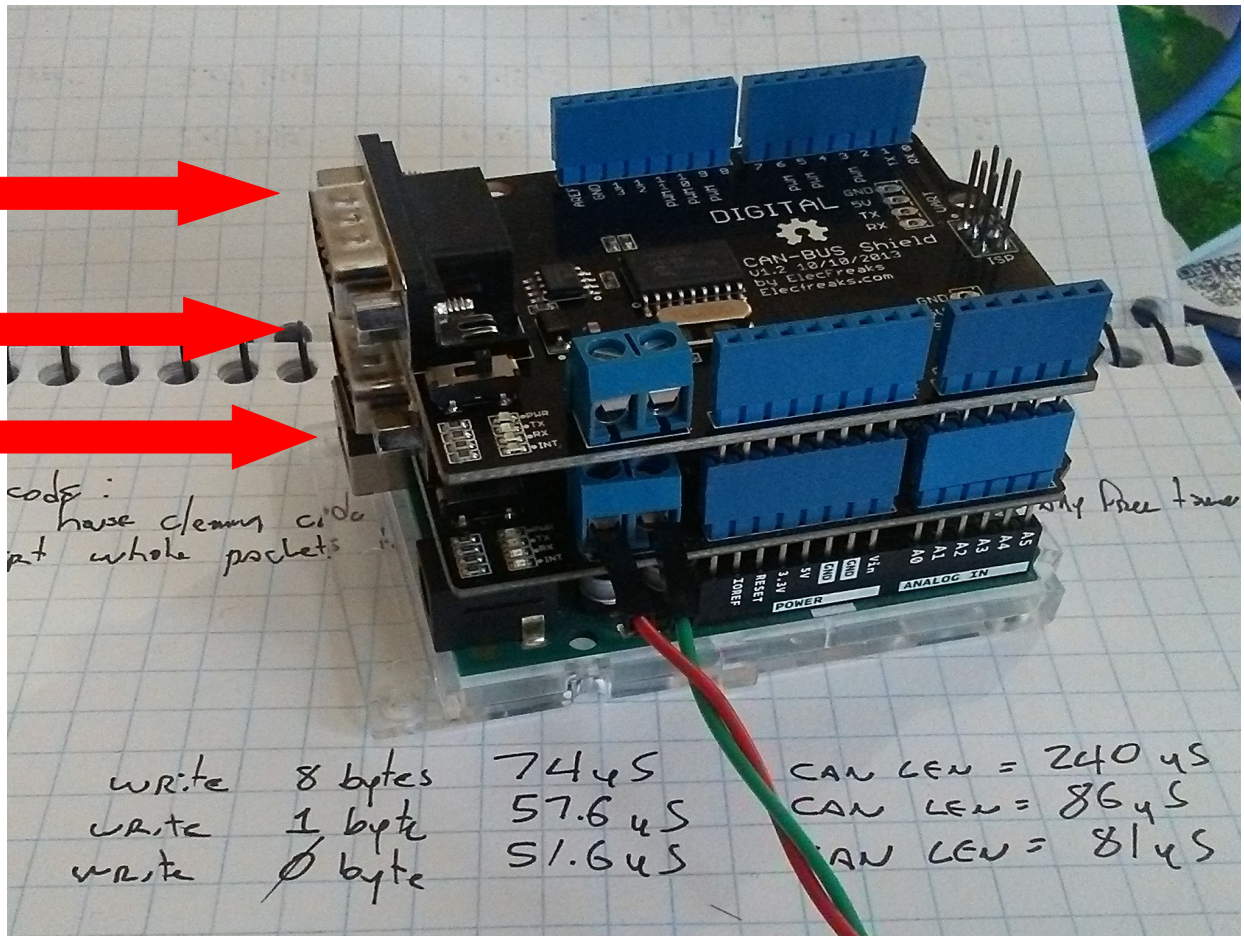
CAN0

CAN1

Arduino Uno
Serial 1E6 BAUD

DB9 Connectors

1 - GND
2 - GND
3 - CANH
4 -
5 - CANL
6 -
7 -
8 -
9 - +12V in

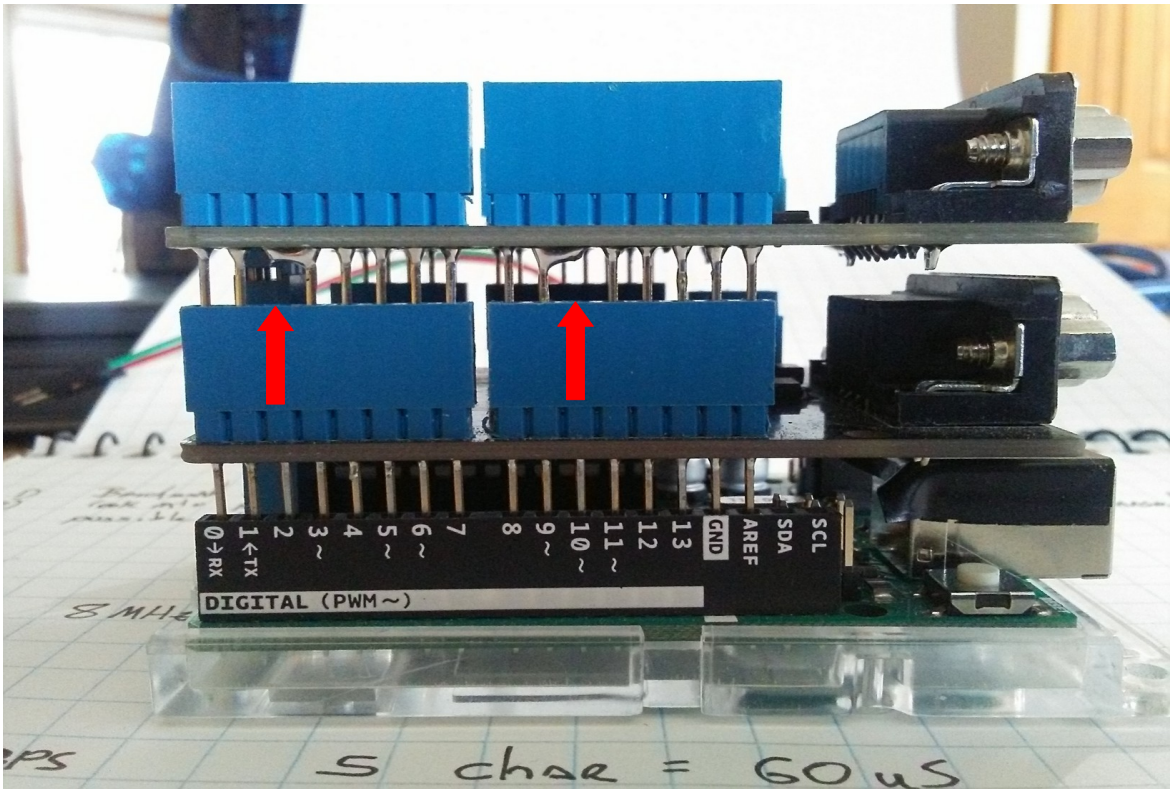


CAN0 (Top)
CS – Arduino Pin 9
INT – Arduino Pin 3
(Note clipped pins
and solder bridges)

CAN1 (Bottom)
CS – Arduino Pin 10
INT – Arduino Pin 2

CAN Modules:
CAN-BUS Shield
V1.2 10/10/2013
By: ElecFreaks
Elecfreaks.com

Sourced from: Amazon
Cost: ~$16.00 each



CANBUS Bridge Settings

Notes:
1. Settings are updated by sending single char values to serial port.
2. Setting are saved to the Arduino EEPROM.

3 output modes available for each CAN channel:

1 = CAN0 no output
2 = CAN0 Source, CAN ID
3 = CAN0 Source, CAN ID, Packet Data

A = CAN1 no output
B = CAN1 Source, CAN ID
C = CAN1 Source, CAN ID, Packet Data

Note: Outputting packet data when the CAN speeds are over 125K will slow down CAN packet transfers. It is recommended to only show the source and CAN ID over 125K. Outputting packet data will cause the RX buffers to overrun in high traffic situations.

Output Mode 1, B:

Source – CAN ID (HEX)

Source 0, 1, 2 = CAN0, number = transmit buffer
Source 4 = no transmit buffers available
Source 5 = invalid ID
Source 6 = invalid length

Source A, B, C = CAN1, number = transmit buffer
Source E = no transmit buffers available
Source F = invalid ID
Source G = invalid length

Output Mode 2, C:

Source – CAN ID (HEX) – Packet Data (HEX)

Source 0, 1, 2 = CAN0, number = transmit buffer
Source 4 = no transmit buffers available
Source 5 = invalid ID
Source 6 = invalid length
Source 9 = filtered packet

Source A, B, C = CAN1, number = transmit buffer
Source E = no transmit buffers available
Source F = invalid ID
Source G = invalid length
Source X = filtered packet

Note: monitor the output buffer, using 2 or more output buffers is an indication of high amounts of traffic

Changing CAN speeds:

6 = CAN0,1 80Kbps
7 = CAN0,1 83.333Kbps
8 = CAN0,1 125Kbps
9 = CAN0,1 500Kbps

Filtering CAN packets:

Notes:
1) Functionality added in V1.0.7
2) These settings are not automatically saved or restored from the EEPROM.

To filet a packet enter '+' and the 3 digit HEX value of the CAN ID.

> **+1F2** will filter CAN packets with the ID 0x1F2.

To remove a filter enter '-' and the 3 digit HEX value of the CAN ID.

> **-1F2** will remove the filter and the packet will be sent through the bridge.

Blocked CAN ID's will still be output to the serial port depending on the output settings selected.
The status byte will be set to 9 for CAN0 and X on CAN1 to show that these packets were filtered.

Saving, Restoring, and Deleting the filter list (functionality added in V1.0.8):

> S = Save the filter list to the EEPROM
> R = Restore the filter list form the EEPROM
> D = Delete the filter list form RAM
> L = List the filter list stored in RAM

Example list of blocked ID's:

> Blocked ID[0]: 1F2
> Blocked ID[1]: 1E8
> Blocked ID[2]: 3F8
> Blocked ID[3]: 2E3
> Blocked ID[4]: 2EC

Example data from a 2008 Dodge Durango radio on CAN0, BCM simulator on CAN1. Settings = 73C.
7 = 83.333Kbps
3 = CAN0 Source, CAN ID, Packet Data
C = CAN1 Source, CAN ID, Packet Data

SRC	ID	B0	B1	B2	B3	B4	B5	B6	B7
A	000	41	00	00	00	00	00		
A	015	55	79	06	FF	00	00		
A	1AF	03	83	00	C0	10	2C	08	00
A	210	02	C8	00	00	00	00		
A	3EC	09	29	2D					
0	3D0	07	0A	0A	0F	0A	0B	FF	
0	3AE	1D	30	00	00	1D	30	00	00
0	3A5	00	FF	FF	FF	00	FF	FF	FF
0	326	00	00	01	0F	07	00	00	00
0	190	01	0D	05	00	10	00	07	
0	18C	00	03	89	21	00	FF	FF	FF
0	0F0	00	00	00	00				
0	416	FC	16	3F	FF	FF	FF	FF	FF
A	000	41	00	00	00	00	00		
A	015	55	79	06	FF	00	00		
A	1AF	03	83	00	C0	10	2C	08	00
A	210	02	C8	00	00	00	00		
A	3EC	09	29	2E					
0	3D0	07	0A	0A	0F	0A	0B	FF	
0	3AE	1D	30	00	00	1D	30	00	00
0	3A5	00	FF	FF	FF	00	FF	FF	FF
0	326	00	00	01	0F	07	00	00	00
0	190	01	0D	05	00	10	00	07	
0	18C	00	03	89	21	00	FF	FF	FF
0	0F0	00	00	00	00				
0	416	FC	16	3F	FF	FF	FF	FF	FF

Note: A DB9 breakout board makes connecting cables to the CAN1 board easier.

Sourced from Amazon