

Building a bi-directional bus buffer that doesn't simply latch at the first LOW!

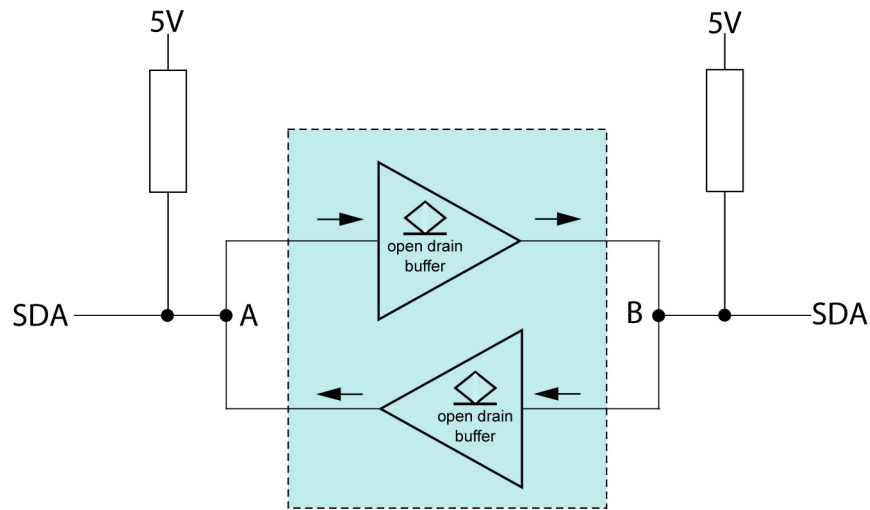


Figure 1: Using conventional buffers with bi-directional I²C signals just causes latching

On the left side of Fig.1, part of an intended 'buffer' device is attached to the SDA wire of an I²C- bus at A. On the right the symmetrical arrangement is connected to the SDA wire of a second I²C- bus at B. The objective is to be able to reproduce the I²C- bus A signals on the second bus B, and also to be able to reproduce the I²C signals from bus B back onto bus A.

Fig.1 shows a conventional attempt to do this, using two simple (uni-directional) open drain logic buffers.

The I²C bus specifications do not allow the bus to be driven towards the HIGH level, so only open collector or open drain buffer outputs may be connected to an I²C- bus. They are able to pull each I²C- bus LOW and it is pulled HIGH only in the normal way, by the passive pull-up resistors as shown.

Here is what actually happens when drive signals are applied to this arrangement. When I²C- bus line A is driven LOW by an I²C device connected to this bus then that bus logic LOW signal appears at the input of a conventional buffer. It has a high input impedance and an open drain output. It correctly generates a logic LOW output on bus B. This is then connected to the

input of a second open drain buffer between B and A. The LOW input at B will then also correctly attempt to drive the SDA line at A to LOW as required. The input A was externally driven LOW but, if the device that pulled that SDA line LOW releases it, the bus at A does not rise because the buffer still holds the line LOW.

The arrangement is essentially a latch configuration and once it is set LOW there is no mechanism to reset it HIGH again. Buffering of this type can only be used if the potential latching loop inside the buffering arrangement can be broken in some way.

Different techniques to prevent latching may be used but there is always some system performance compromise involved. The objective is to find techniques that avoid latching but minimise the consequences of their compromise on the system.

Another important point to note is that, within the constraints of simple binary logic, with just one HIGH and one LOW level, there is no way to build a buffer device that is truly bi-directional. It is simple to build a buffer than can be switched so that it can pass signals

in each of two directions – but not in those two directions at the same time.

Because the ‘building blocks’ for an I²C buffer will be uni-directional, the challenge is to find ways to split the bi-directional I²C signals into uni-directional data streams and recombine them again.

Most of the solutions use analog level techniques and are covered in detail in the descriptions of each buffer class. Refer [TR005: Non-isolating bus extenders](#), [TR006: True voltage followers type buffers](#) and [TR007: Special logic low I/O levels type buffers](#).

They, directly or indirectly, introduce additional logic switching levels and use these to determine the origin of the signal that is driving the bus LOW.

For example, the buffer with its input at A can be specially designed so that its input logic switching level is always slightly lower than the level to which the other open collector buffer’s output can pull down bus A.

Then when this buffer arrangement is driving the I²C-bus at A to LOW that LOW level will not be recognised by the buffer sensing the I²C bus at A and it will not drive B to LOW. The buffer’s output LOW at A might be set at 0.7 V and the buffer’s input switching level set below that, say at 0.5 V. The 0.7 V driven by the buffer is compatible with all I²C driver devices connected at A. They are required to recognise any voltage less than $0.3V_{CC}$ as being LOW. This buffer can recognise other I²C devices connected at A because they can be arranged to pull the bus below 0.5 V. The buffer’s input LOW is not ‘compliant’, because the I²C LOW requirement ranges up to $0.25V_{CC}$, but this compromise on logic levels provides one solution to the latching problem.

Designing an I²C system? Visit www.bus-buffer.com for more information or to contact us.