

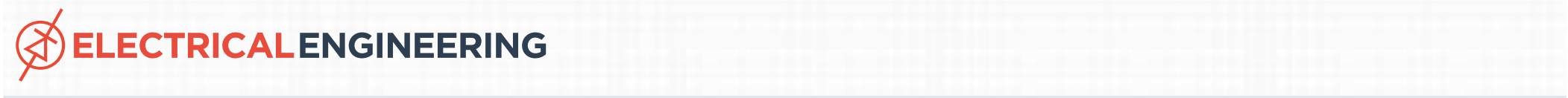
Electrical Engineering Stack Exchange is a question and answer site for electronics and electrical engineering professionals, students, and enthusiasts. It only takes a minute to sign up.

Sign up to join this community

Anybody can ask a question

Anybody can answer

The best answers are voted up and rise to the top



How to resolve I2C address clashes?

Asked 12 years ago Modified 3 years, 8 months ago Viewed 54k times

45

I want to connect multiple I2C slave devices to a micro controller all on the same set of pins but the I2C devices all share the same address. The addresses are fixed in the hardware.

Is there any way to connect multiple devices with the same address?

Perhaps some kind of I2C address translation module with each device with an configurable address so I can assign my own addresses to each one.

serial

i2c

Share

Cite

Follow

asked Oct 11, 2010 at 14:40

Simon P Stevens

636 ● 1 ● 5 ● 10

8 Answers

Sorted by: Highest score (default)

26

There is nothing built into I2C to do this, normally slave devices will have some externals pins that can be set to 0 or 1 to toggle a couple of the address bits to avoid this issue. Alternatively I've dealt with a few manufacturers that have 4 or 5 part numbers for a part, the only difference being its I2C address.

Most devices have specific hardware that handles the I2C communication, that is the slave ACK is in hardware so you really can't hack around it.

As for the translation module, you could buy some \$0.50 PIC's with 2 I2C buses and write some quick code to make them act as address translators i guess.

Share

Cite

Follow

answered Oct 11, 2010 at 14:56

Mark

11.5k ● 30 ● 38

Thanks. Yeah, these devices do have an address select, but only between two addresses and I want to connect 5+ devices, so I'd still end up with clashes. I hadn't thought of using a PIC. That should work. Is there nothing off the shelf that does this kind of thing? – Simon P Stevens Oct 11, 2010 at 15:19

13 NXP makes a bunch of multiplexers / switches for I2C, you may be able to rig something up out of those: [ics.nxp.com/products/i2cmuxes](https://www.nxp.com/products/i2cmuxes) for instance you could create in your case 3 sub branches, each with 2 devices on it, and use one of NXP's switches to achieve your goal. – Mark Oct 11, 2010 at 16:12

Great, that's exactly the kind of thing I was looking for. I just didn't know the name. Thanks. – Simon P Stevens Oct 11, 2010 at 18:11

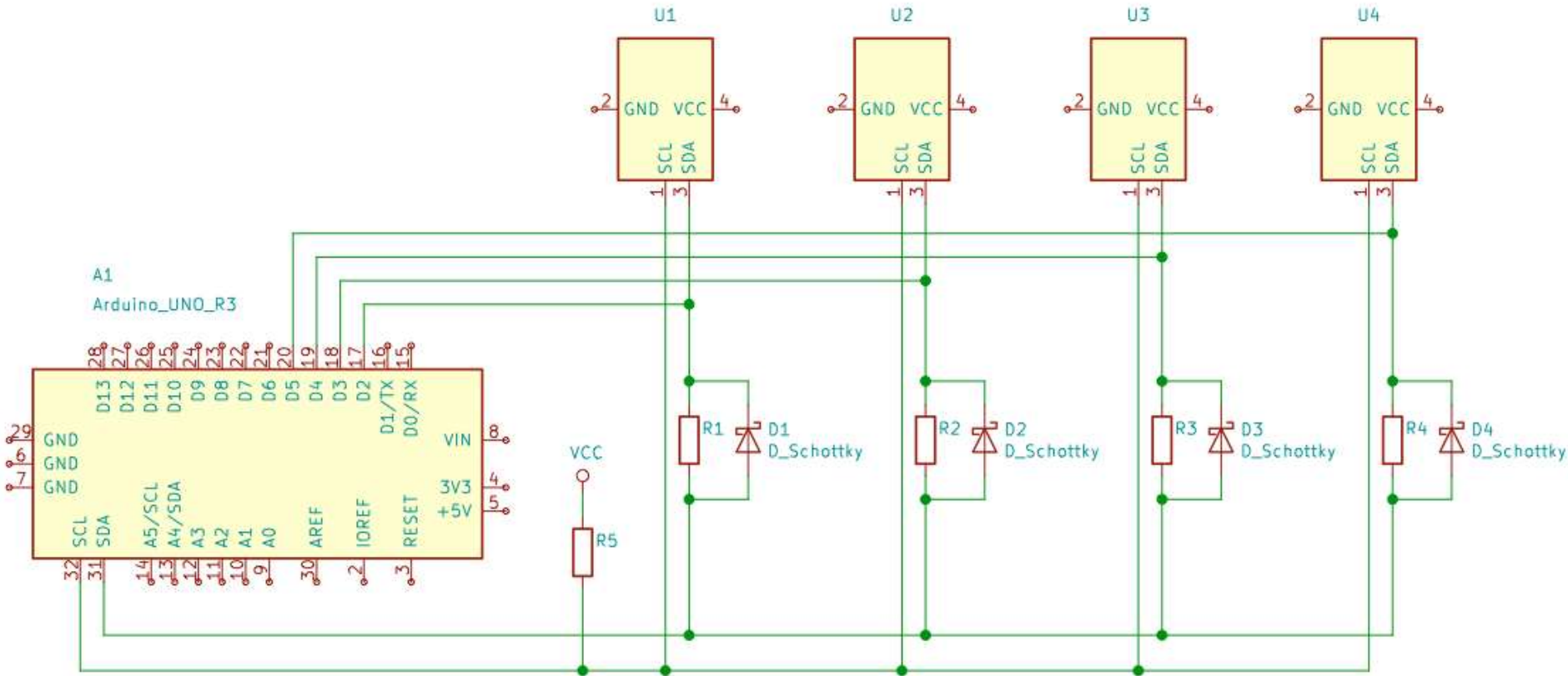
1 This answer is 7 years old, so I believe that it could have been the best at the time it was offered. For others coming through now, however, some of the much newer answers (currently lower ranked on the list) offer potentially better approaches based on newer components that are now on the market. – Brick Apr 21, 2017 at 15:59

13

I've just run into this problem with multiple I2C devices with a fixed address. Our solution was to use I/O lines on the microcontroller to force the SDA lines high on the devices that we *don't* want to address, while the I/O line for the device we're targeting is set as input (high impedance). This means that only the targeted device matches it's I2C address and the others ignore any subsequent data.

https://electronics.stackexchange.com/questions/5096/how-to-resolve-i2c-address-clashes

1/4



The resistors on the SDA line for the inactive devices end up acting as pull-ups for the bus, so the exact value will depend on how many devices you have and what pull-up you need for your bus. So if you choose 10K resistors, then 3 inactive devices gives a 3K3 pullup.

The schottky diodes ensure that the device can still pull the SDA line low enough when transmitting data back to the host.

Share Cite Follow

answered Aug 22, 2018 at 6:48


 **Peter Gibson**
1,790 ● 1 ● 12 ● 14

- I appreciate that you've followed up and posted this. This is a pretty ingenious solution, I'm certain it will be helpful to others. – **Simon P Stevens** Aug 22, 2018 at 8:23
- A nice niche that can be used on *some* applications. I like it a lot. – **Harry Svensson** Aug 22, 2018 at 11:07
- isnt this a workaround for wrong HW design ? What if we dont have extra IO pins to spare ? – **a k** May 21, 2020 at 3:05
- 1 This is the least expensive solution. Our product has two equal sensors and has no budget for any additional IC. Here we only add two diodes and one resistor. Thank you! – **EmbeddedGuy** Jun 27, 2021 at 8:20

There is now an answer- Linear Tech has the LTC4316/17/18 series of address translators. They are relatively new, and availability is uncertain.

Share Cite Follow

answered Jun 25, 2015 at 0:02

 **user3608541**
111 ● 1 ● 1

Very interesting component. Most I2C devices have 2 fixed addresses and this can LTC4316 could be potentially double the addressing in a reasonable cost. – **Mehrad** Feb 16, 2016 at 6:53

If none of the I2C devices use clock stretching (handshaking), and if you're bit-banging the I2C master, a simple hack is to have some of the devices swap the clock and data pins. During the transmission of a byte, the device which has the clock and data pins swapped will see each "0" bit as a non-event (data rising and falling with no clock) and will see each "1" bit as an I2C stop and start (clock rising while data is low, fallowed by data rising and falling, followed by clock falling). Intentional stop and start conditions for one device may be seen as data bits by the other, but unless one device has an excessive number of start and stop conditions between "1" bits, it would be unlikely that any device would

"accidentally" see a start condition followed by eight data bits without an intervening stop condition.

Share Cite Follow

answered Dec 12, 2011 at 20:14

supercat
44.9k ● 2 ● 81 ● 140

6

I'm not downvoting, but this seems a bit risky to me. My experience with I2C is that it's noise-prone enough with just the usual connection. You do, however, use the word "hack" and mention the caveat "if none of the i2c devices use clock stretching", so if it can work for someone, then more power to them. – [Jason S](#) Dec 12, 2011 at 22:12

Several manufacturers offer I2C bus multiplex- and switch ICs.

6

A mux can activate one channel at a time; a switch can enable multiple ones in parallel.

Check for example the offerings of [NXP](#), [TI](#) and [Maxim](#).

For experimentation, Adafruit has a [TCA9548a board](#).

If you have 8 target chips with identical addresses, select an 8-to-one MUX. Before accessing any of the target chips, configure the MUX to activate the correct I2C bus.

Advantages

- Requires no programming (vs microcontroller based approach)
- Can support the I2C features and speeds you need (vs regular analog / digital bus muxes). For example, a regular (non-I2C) MUX will not pass general call addresses to all its channels.

Share Cite Follow

edited Dec 8, 2017 at 10:20

answered Mar 29, 2017 at 14:18

florisl
291 ● 3 ● 5

I had two TCS3414 color-light sensors that I wanted to compare (The FN and CS packages, which have different filters). The I2C address is hardwired. After looking at how I2C works in terms of the SCL(clock) and SDA(data) lines, it seemed that turning off the SDA line would prevent the chip from getting a start or stop bit and thus leave it dormant. So used a CMOS analog switch (4066B) to switch on or off the SDA line to each device. This worked just fine for switching between the two devices. I know it's a hack, and the PCA9548 would be much better, but I didn't have one handy.

Share Cite Follow

answered Feb 18, 2013 at 4:30

David Hill
51 ● 1 ● 1

1

Actually, this isn't a hack at all, and I'd argue this should be the accepted answer. I've seen this used in several commercial products, and I can't think of a better solution (unless you have no GPIO available and thus need a pure I2C solution such as the I2C-specific muxes). Good ol' analog muxes have plenty of bandwidth and are crazy-cheap. – [Jay Carlson](#) Jan 16, 2017 at 6:10

I'd consider using [bus switches](#) to multiplex the I2C bus among the devices with conflicting addresses. Bus switches are very low capacitance and resistance, and unlike buffers/drivers, they are true switches that connect or disconnect two circuit nodes.

5

Bus switches usually have one odd characteristic, that doesn't matter for I2C because it uses open-drain devices: a bus switch has low on-resistance when tying together voltages near 0 (Vss), but the resistance rises dramatically as the voltages approach the power supply Vdd. (This is because they're basically MOSFETs with gate voltages at the power supply when they turn on, so as the switched voltages approach Vdd, the available Vgs is much lower)

Share Cite Follow

edited Jan 31, 2019 at 7:07

stonecrusher
123 ● 4

answered Dec 12, 2011 at 22:16

Jason S
13.9k ● 3 ● 40 ● 66

1

The link is broken. [fairchildsemi.com/product-technology/bus-switches](#) works better. – [florisl](#) Mar 29, 2017 at 14:12

Use a simple demux chip (e.g. 74HC139 afaik) and connect I2C CLK pin to input (since I2C CLK pin is output only). Use GPIO pins to control desired output. Then I2C data pin can be shared between all slaves.


0

Share Cite Follow

answered Dec 13, 2011 at 6:41

Tomas D.
1,011 ● 5 ● 7

6 SCL is *not* output only. A slave may [stretch the clock](#) if it needs to slow it down. – [stevenvh](#) Apr 7, 2012 at 15:59

You can use an analog multiplexer (which is bidirectional) but a decoder may not work for the reason stated by [stevenh](#). If you use multiplexer you will need a weak pull-up on the slave side to ensure it is kept inactive. Also only change the multiplexer selection when the bus is idle. – [Kevin White](#) Jun 25, 2015 at 2:09 

While it's not a general solution, it can be done with most ICs that don't use an internal microcontroller for the I2C interface. Devices that are safe to use can usually be determined by looking at the pin descriptions or the block diagram in the respective datasheets. If SCL is configured as just an input, this would work. – [Benedikt M.](#) Jan 26, 2021 at 3:09
