Introducing the P82B96 bus buffer...

Philips I²C Bus

- *extending its reach*

Let's make things better

PHILIPS

Presentation revised May 2001. Reporting of any errors found, or suggestions for improvement, to philip.tracy@philips.com will be appreciated.

Objective: Introduce the 82B96 chip, reasons for its development and some applications for it.

The introduction of a viable 'buffer' for I2C signals means that I2C no longer needs to be considered as an interconnection for ICs on a printed circuit board.

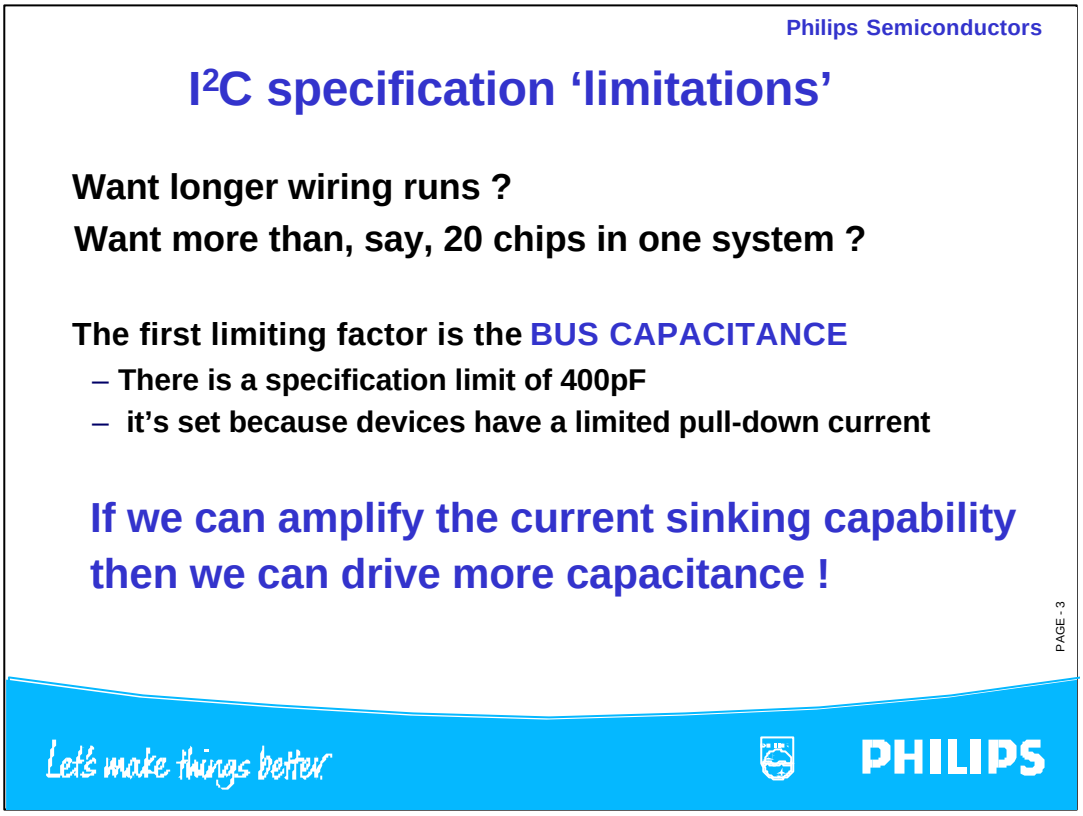
I2C can now interconnect ICs or modules over long cables, across opto-isolation barriers, or even over RF links.

What are the I²C fundamental limitations ?

...There are some implied in its specifications,

but its greatest limitation is just the

system designer's imagination !



The I2C bus lines are passively pulled up using resistors (or sometimes current sources).

A device wanting to pull the bus lines low needs a sinking capability greater than the pull-up current.

So, for inter-operability, the specifications need to define a minimum current that all devices must be able to sink.

This is set at 3 mA for I2C and 350 uA for SMBus and, in turn, sets a lower resistance limit on the pull-up resistor. In a 5V I2C system it is about 1.6k ohms.

The pull-up resistance together with the bus capacitance determines the rise-time of the bus wires. I2C sets a limit of 1 microsecond for the rise time defined between two voltage levels. It can be met in 5V systems if the R-C product of the bus pull-up resistor and total bus capacitance is less than about 1.6 microseconds.

System design limitations

Want to interface to the hardware of another bus ?

- **Busses like RS485 and CAN would be attractive for long distances, but no-one offers a converter** (till now !)
- **Why ?? It's 'impossible' !**

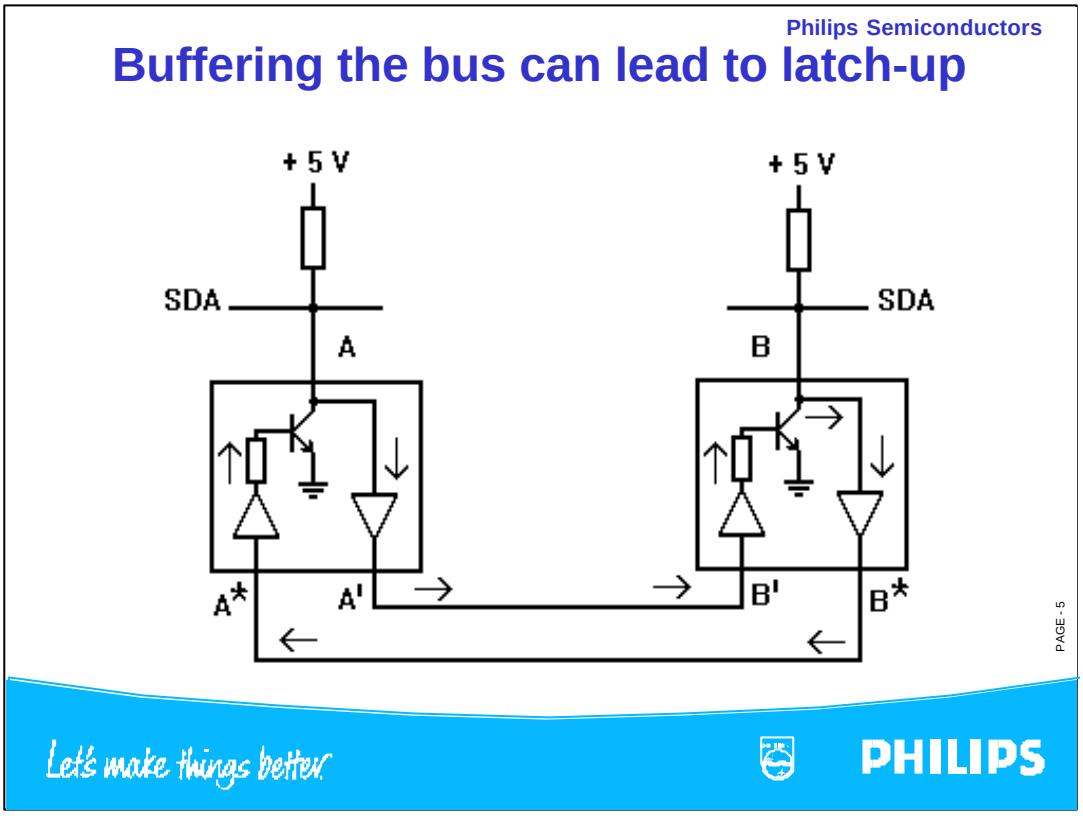
Want to Opto-Isolate parts of your I²C system ?

- **Magazines are filled with attempts - all have stability problems**

Buffering or isolation requires a bi-directional, unity-gain amplifier – one that doesn't latch-up or oscillate !

It still remains 'theoretically' impossible to buffer I2C signals with logic chips using the conventional logic signal levels used by logic chips.

Solutions involving buffers that are 'semi-bidirectional' – ones that switch between two unidirectional modes – introduce 'glitches' onto the I2C bus that cause non-conformance to the bus specifications.



A ‘buffer’ device is attached to the SDA wire at A.

When A goes low it generates a low output to some new and different bus at A'.

This is transmitted over some path to an equal buffer at B.

The low input at B' drives the SDA line at B to low as required.

But the buffer at B is sensing the SDA bus and will then output a low at B*.

This is transmitted back to the first buffer as an input at A*.

The buffer at A now drives its SDA line low.

Even when the device that originally pulled A to low releases the SDA line the buffer just holds it low.

Buffering of this type can only be used if the loop around the two buffers is broken in some way.

In P82B96 the problem has been overcome by introducing two different logic ‘low’ voltage levels, and keeping both those levels inside the range specified as ‘low’ for the I^2C.

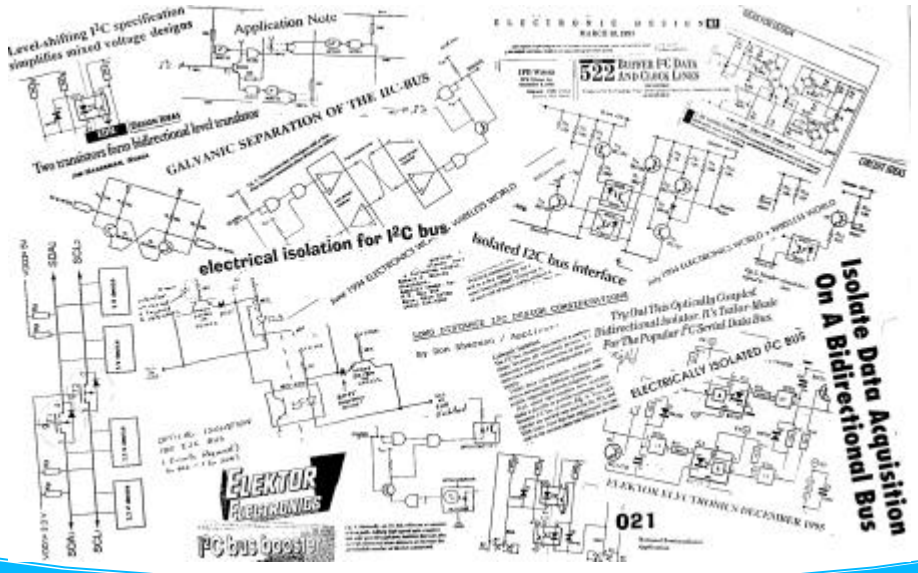
P82B96 will monitor the voltage levels on bus A and generate a logic ‘low’ signal A' (data to be transmitted to a remote node) only if the bus is below 0.6 V.

A logic ‘low’ on its input A* (data received from a remote node) will cause the I^2C bus at A to be pulled down to 0.9 V.

Both 0.6 V and 0.9 V are just a logic ‘low’ for ordinary ICs on the 5 V I^2C bus, the specified maximum low level being 1.5 V.

But a drive signal arriving at A* will not be re-transmitted at A' – bus latch-up is avoided !

Many designers have tried to do it !



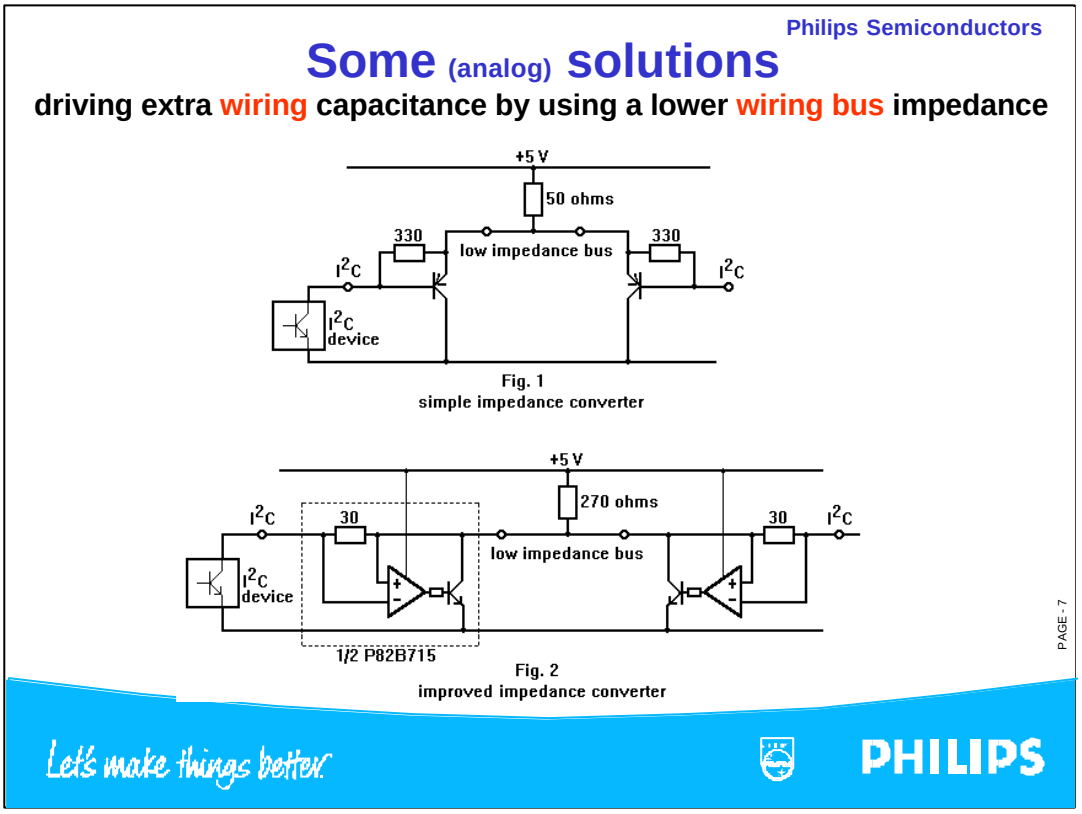
Let's make things better

PHILIPS

The press is filled with attempts to ‘buffer’, ‘extend’, ‘isolate’, or level shift.

The market demand is clear.

Almost all of these circuits produce false bus signals.



Simple transistor impedance converter.

A 'stone age' solution, but one of a very few solutions that reasonably complies to bus standards. No 'glitches'. A bus 'low' level of typically 0.8 V, same as the Sx/Sy side of P82B96. It has wide usage and built up a market demand for P82B96.

A transistor gives great gain. Here 50 ohms can be driven - within the I2C chip ratings.

But that gain can also cause delay problems. Imagine the right hand I2C side was changed. Delete the 330 ohms emitter-base and put 200 pF to ground. That capacitor has the emitter-follower current gain action working to hold the bus low. To pull down 10 mA in the bus resistor needs less than 50 uA of base current flowing into the capacitor. In effect, the bus resistor is multiplied by the transistor gain. The 200pF has an effective pull-up resistor of 50 ohms multiplied by the hFE – say 250. The bus pull-up time-constant is 50*250*200 = 2.5 usec.

Secondly the I2C chip has to drive EVERYTHING connected in the whole system and any interference signals from the long wiring have a path back into the driving chips.

P82B715.

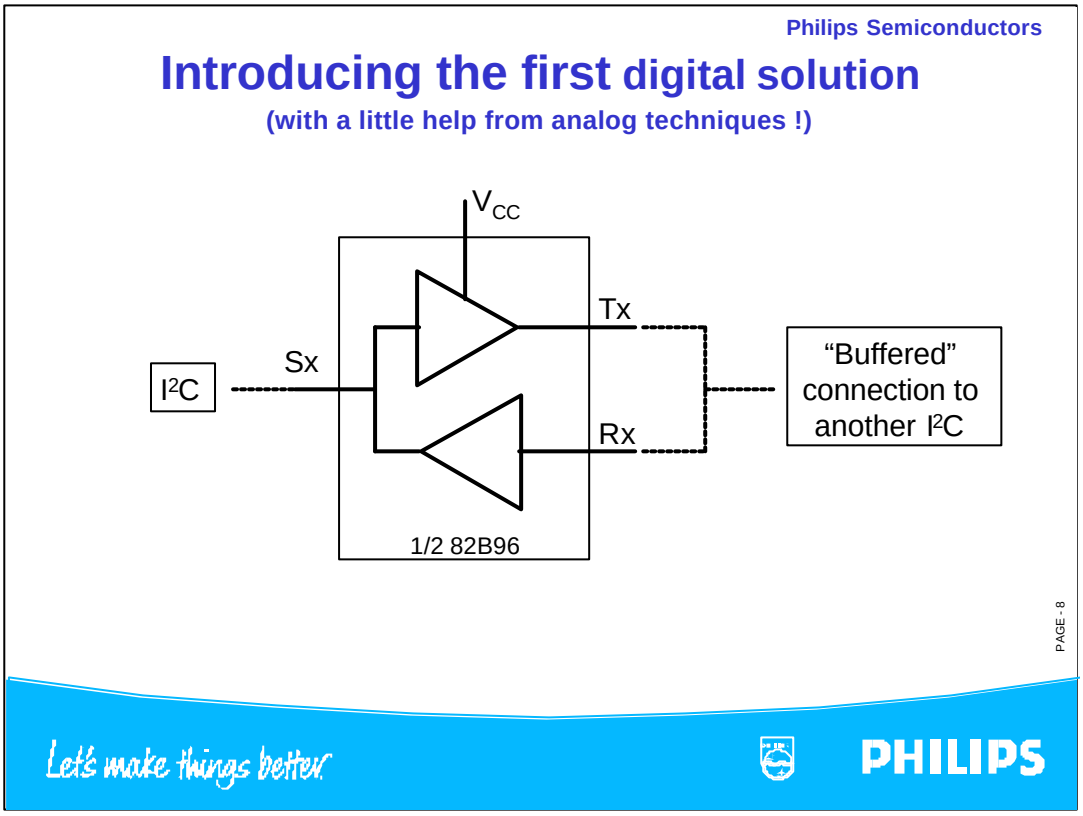
This is the simple transistor solution, but without the emitter-base voltage drop. So full logic swings from 0 V to 5 V on both sides. The transistor 'gain' problem is eliminated by setting the IC current gain to 10. It is still just a 'voltage follower', there cannot be any logic level shifts.

The bus rise-time delay problems that can occur with the single transistor, leading to system instability once the inductance of wiring is included, are effectively eliminated.

Transients are clamped to Vcc/ground by large internal transistors/diodes, protecting I2C chips.

Neither the transistor nor P82B715 is really an interface device. They don't truly 'buffer' or change the bus signals, they only amplify (or drive) in one direction. In the other direction the bus signal passes PASSIVELY through a resistor. They can't change the bus voltage levels.

P82B96 provides true 'buffer' action. The I2C chips only have to drive its high impedance, low capacitance input. P82B96 allows an increase in noise margins on the low impedance bus side by converting up to 12V or 15V logic levels, much better suited to long wiring



For PRACTICAL purposes 82B96 offers the characteristics of an ideal buffer.

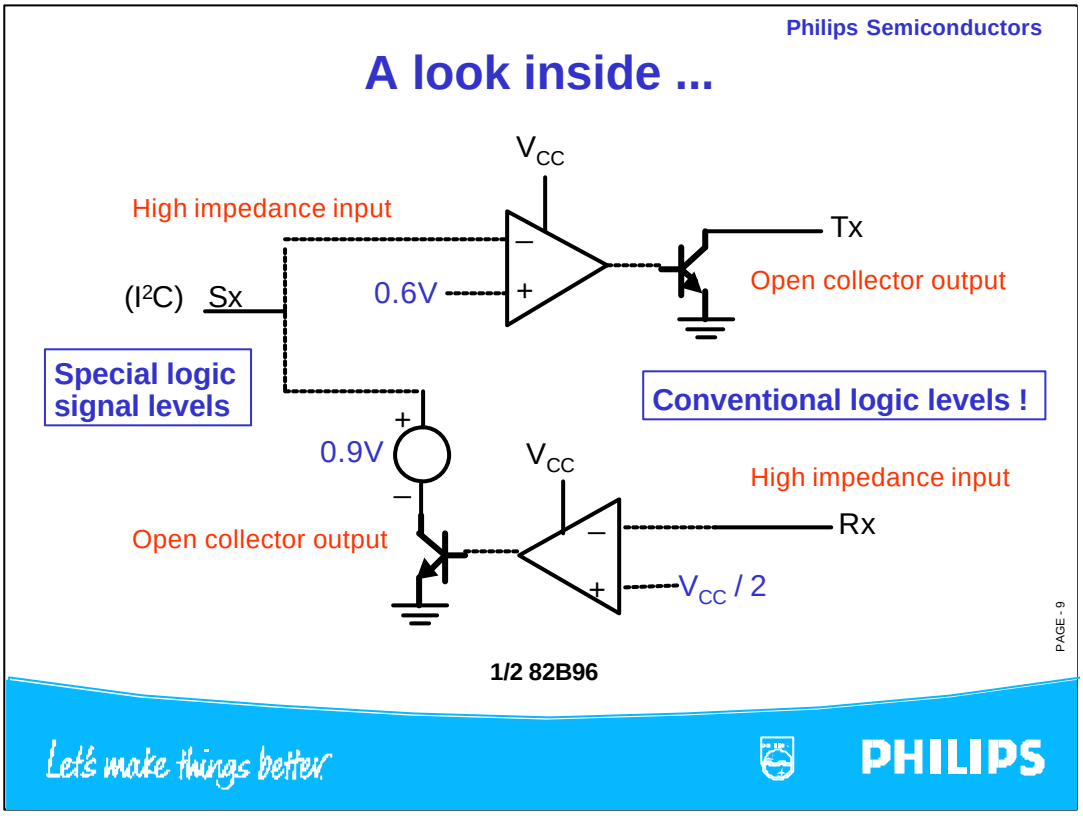
The I2C bus on either side can fully communicate with another I2C bus on the other side of the chip. All features of the I2C protocol are supported.

The two separate I2C busses only 'see' an additional loading of the input impedance of the buffer chip – a few pF and a microamp !

It is allowed to have many 82B96 chips connected to the I2C bus on the right hand side (the side with Rx and Tx linked) without any restrictions on I2C operation.

So MANY different I2C busses (each with their allowed 400pF) can now be interconnected to make one large I2C system.

Better still, the I2C bus on the right hand side has enhanced features and can also be simply configured as a 'special' I2C bus with 4000pF loading allowed !



It's not necessary to understand the 'inside' to simply use 82B96 in systems, but a little understanding can help.

We apply unconventional logic voltage levels on ONE side to prevent latching.

On the right hand side the logic levels are just standard 'CMOS' logic levels.

The threshold for change from logic high-to-low is half the chip supply voltage.

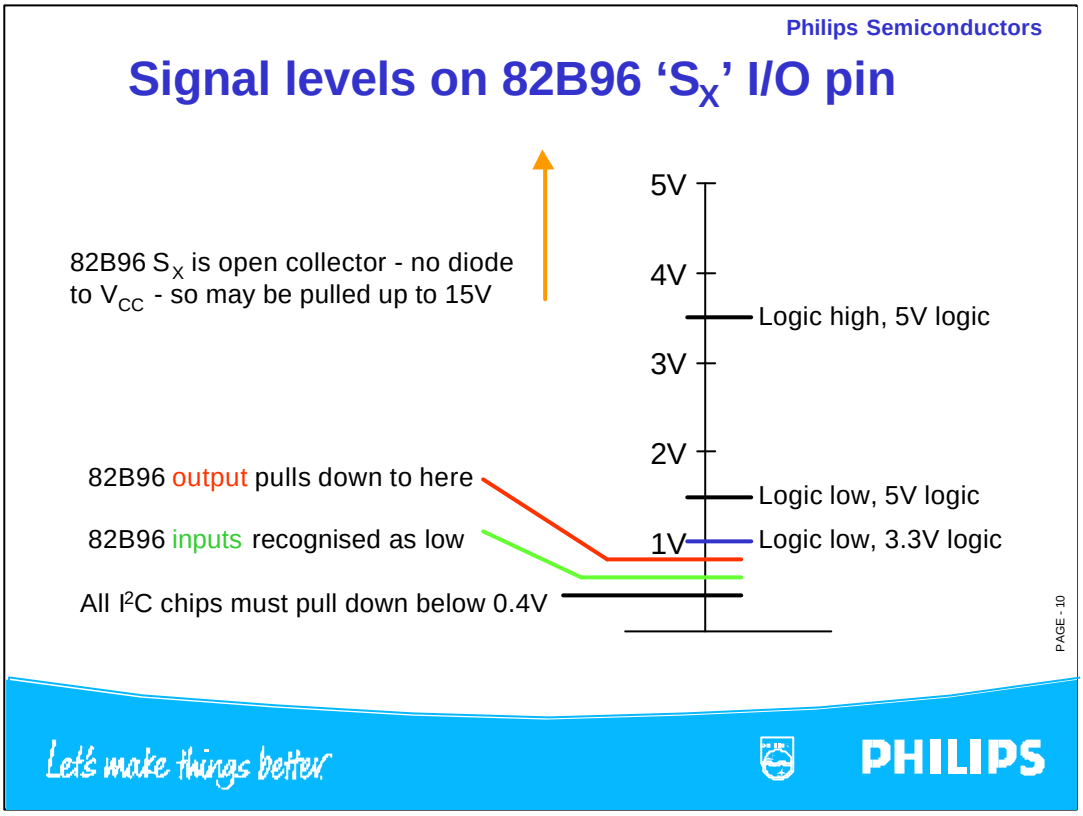
It is allowed to directly link Tx to Rx and form a new, perfectly standard, I2C bus interface with conventional properties.

But on the left side the logic 'low' input level is 0.6V and the logic 'low' output voltage is 0.9V.

We add one more advantage: While the Tx pin can be a normal I2C output, we also gave it extra sinking capability.

It's guaranteed sink capability is 10 times the normal. So 30 mA static, and around 100mA dynamic.

This permits a non-standard bus with 4000pF and 10 times lower pull-up resistance to be made.



Notice that the 82B96 Sx pin pulls the bus down to 0.9V, which is below the level required to guarantee it is recognised by other connected chips as a 'low' on either a 5V or 3.3 V CMOSsystem.

But 0.9V is not recognised by 82B96 as a logic 'low' input, so it is not transferred to the Tx output - this is how latching has been avoided.

Again, all external I2C chips are required to pull the I2C bus down to a static 'low' level of 0.4V, so when an externally connected chip pulls the bus 'low' it is recognised at the Sx input and transferred to the Tx output.

The Sx input logic threshold and output sink voltage are independent of the 82B96 supply voltage. The supply range specified is from 2V to 15V.

Notice now the ONLY compromise we had to make to (almost) achieve that impossible 'perfect' buffer is that we cannot allow connection of multiple 82B96 chips on their 'Sx' side because it is using these special logic voltage levels.

The I2C 'low' signals cannot pass from one 82B96 to another one via these Sx logic signal levels -- for the same reason that prevents an 82B96 from latching.

(Of course it is possible to connect the Sx sides of multiple 82B96 chips provided ALL the bus MASTER chips are also connected to this node)

Philips Semiconductors

What does your design require ?

need to…	Solution can be…		
	82B96	82B715	Fets
Voltage level convert I2C bus, low capacitive loads only	yes	-	yes
Convert I2C bus levels, including heavy R or C loadings	yes	-	-
Convert I2C bus voltage levels to gain noise immunity	yes	-	-
Drive long wiring between I2C nodes	yes	yes	-
Add more ICs than I2C capacitance spec allows, all on 1 PCB	no limit	up to 100	-
Add more ICs and also drive long bus wiring	yes	-	-
Include galvanic isolation of bus (opto-isolate some sections)	yes	-	-
Convert I2C bus signals to another hardware standard	yes	-	-
Interface standard I2C to the low current SMBus (350 uA)	yes	-	-
Operate on low supply voltages	> 2 V	>4.5 V (>3)	> 2 V
Operate on low supply current	< 1 mA	16 mA	zero !
Disconnect section of I2C if a chip's supply fails	yes	-	yes
Send I2C signals over radio or infra red links	yes	-	-
Provide ESD protection for ICs in I2C system with plug-ins	yes	yes	-

PAGE - 11

Let's make things better PHILIPS

Driving long wiring: 82B715 has a limit of about 3000pF wiring capacitance. That scales to 300pF, allowing another 100pF for the total of all the ICs in the system. Total < 400pF.

Driving more ICs: 82B715 can be applied in an unusual way to allow up to about 100 I2C chips in a system. The total system capacitance, including wiring, can be about 1500pF. Request slide '100 ICs on one I2C bus'. When using 82B96 there is no limit on connected devices or the capacitance of chips or wiring.

Changing Bus hardware standards. It is possible to send I2C signals, in I2C format, over the hardware of busses like CAN or RS485. Conversion to CAN twisted pair signalling can be via 82C250. Note it takes two twisted pairs to carry the two signals SDA/SCL.

It is also possible to use a Microcontroller to convert the signalling format to the signals used with other busses, but our simple application diagrams do NOT show or suggest that.

SMBus is an application of I2C proposed by Intel/Duracell and several others. The logic levels are different and the maximum static sink current is 350uA. It was originally proposed for applications in rechargeable battery management but has become the standard for system management in PCs.

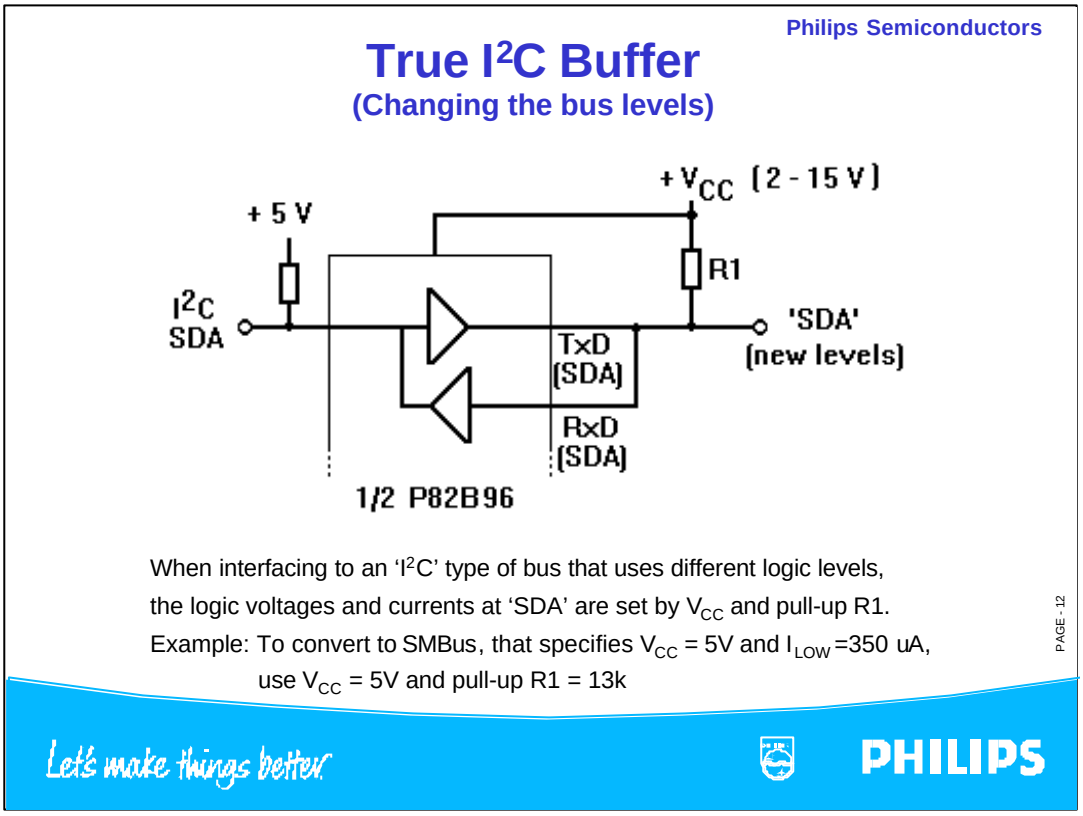
Providing ESD protection. The large bipolar transistors and substrate diodes inside P82B715 will carry large transient currents without damage. That action can be used to protect other connected I2C chips in 'plug and play' systems where different parts of an I2C system are connected via plugs/sockets.

P82B96 protects in an even safer way, any one side is actually isolated from the other side.

Even if the 82B96 is destroyed on one side, the I2C chips connected at the other side are separated by a good barrier. Its 15V tolerant I/Os allow higher voltage zener clamping, meaning lower capacitance

Low supply voltage. 82B96 is guaranteed down to 2 V supply. 82B715 has full performance guarantees down to 4.5 V. Typically it has full performance down to 3V but the guaranteed performance at 3 V is slightly reduced. The current data does not show the 3 V guarantees but a sheet with guarantees can be supplied on request.

The Fet solution limit depends on the gate-source/gate drain threshold guarantees. 3 V is no problem and typical operation should be ok down to 2 V.



It is allowed to simply link the RxD and TxD pins to form a new bi-directional bus with all the same properties as any standard I2C bus.

The new bus could be just another standard 5 V I2C bus, with the chip function being simply to allow more ICs to be connected in the system without exceeding the 400 pF specs.

A total of 400 pF would be allowed on each side of the interface. PCA9515 can also be used to link (just) two standard 400pF busses like this.

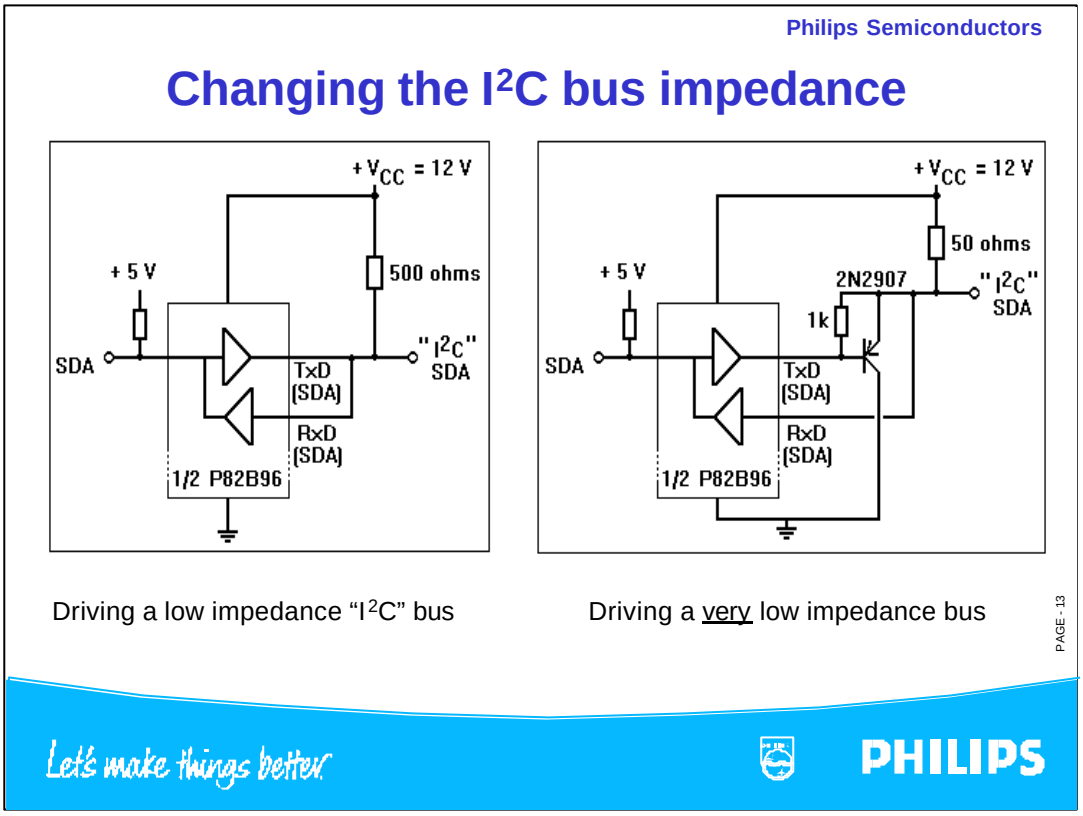
It IS allowed to interconnect, say, 10 of these interfaces together on the 'new' bus formed by the linked TxD/RxD pins and thus form 10 new I2C nodes on the other side of those interfaces. Each new node may be loaded with 400 pF.

Such a system now permits 10 x 400 pF in total and the system has full I2C performance.

Note however that it is NOT allowed to connect more than ONE interface to any one I2C node using its 'left hand' input because it is NOT possible to pass data from one interface chip to another via those terminals - this restriction is imposed by the special 3-level logic used there to prevent latching.

The logic levels used on the RxD pin are made proportional to the supply voltage, so that those logic thresholds will truly 'match' the bus voltage to which they are connected. e.g. if the Vdd = 3 V then the guaranteed logic low will be 1.26 V and the high will be 1.74 V (nominal switching level = 1.5 V, half the Vdd supply.)

TxD is just open-collector with 'low' = 0 V, and 'high' set by the pull-up to the supply.



On the left we use the 30mA sink capability of P82B96 to directly drive a 500 ohm bus with 12V logic swings.

These levels / impedances are suitable for transmission over conventional twisted pair telephony wiring.

One 'pair' could be SDA plus ground, a second 'pair' could be SCL plus Vcc.

Probably there will be several different 12V supplies and the ground potential will be different at different places. Then the 500 ohms is best distributed around the system, say 5k ohms pull-up at each bus node in a system with 10 nodes.

The logic switching level of half-rail (6 V) means that any interfering noise, voltage drops in the wiring, or differences in local ground voltage must be able to cause a 6V signal error before bus logic errors occur.

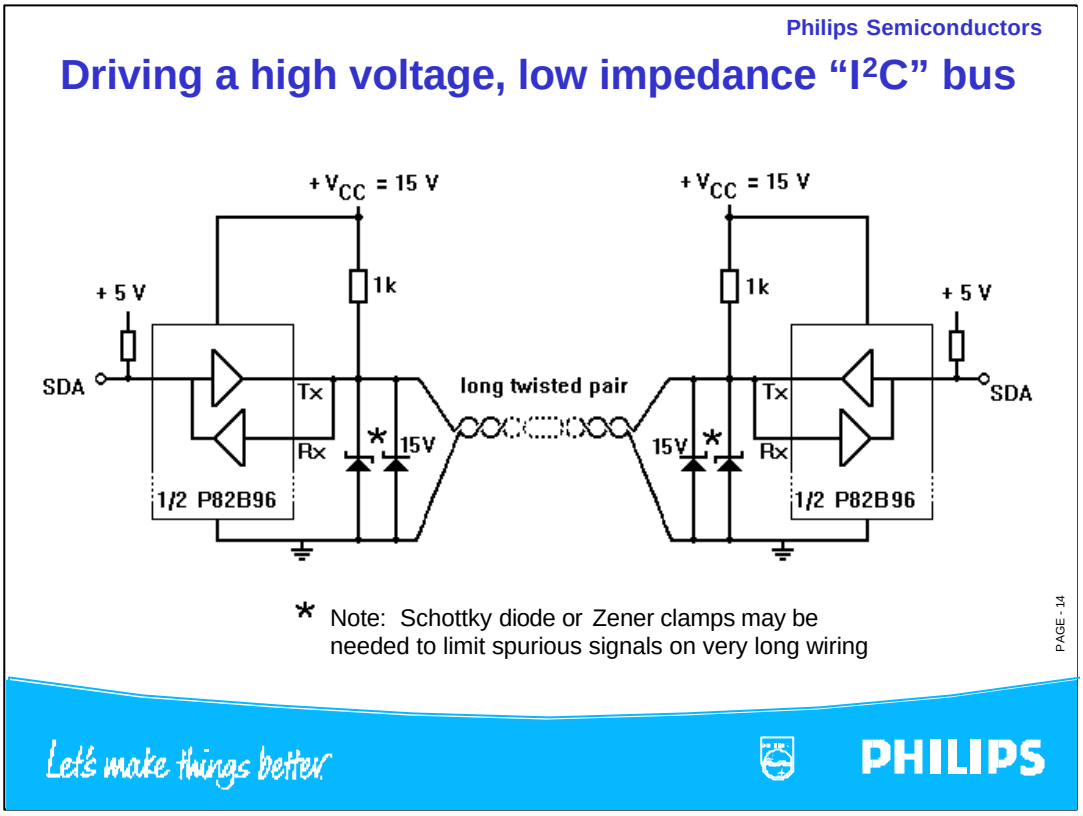
On the right we show a really low impedance bus - 50 ohms.

Since the P82B96 can't directly drive the necessary 230 mA we add a PNP emitter-follower transistor.

Note that the high current flows from the pull-up resistor through the transistor emitter and collector to ground. Only the transistor base drive current flows in the open-collector output stage of the IC.

The IC is specified to sink 30 mA, so with a transistor gain of less than 10 we can sink the 230 mA required in this circuit.

Like all I2C chips, that must be capable of sinking more than their static 3 mA during the time the bus is falling, this circuit needs additional dynamic sink capability to charge any capacitance associated with the wiring. P82B96 can provide typically 100mA base drive, the 2N2907 is rated to 800 mA peak collector current. A transistor with even higher current rating device can be used if necessary.



Creating a new, long distance, bus with the same features as I2C.

Here 82B96 is applied to convert a standard 5V/3mA I2C bus, probably using 4.7k ohm pull-up impedance, to a new bus running at 15V logic levels and 500 ohms impedance (30 mA). There can be many P82B96s connected to the bus, not just two. The pull-up could be spread over the system, at each connection just make pull-up = (500 ohms X number of connections).

That low impedance bus can be run using typical twisted pair telephone wiring for distances up to at least 100 metres and even at clock speeds up to 100kHz.

One limiting factor becomes the speed of light ! The bus signals travel about 1 metre in 5 nanoseconds in plastic insulated cables. The 'acknowledge' signal from a remote IC must be received while the bus clock is still low during each 9th clock pulse.

That is, one IC sets the clock line (SCL) low at the end of the 8th clock pulse. That 'low' travels along the wiring to a remote IC. When it arrives that IC needs to generate a data bus (SDA) 'low' to acknowledge. That may take some time, although it is only specified for a few chips - such as memory chips. PCF8570 and PCF8583 specify 3.4 usecs maximum.

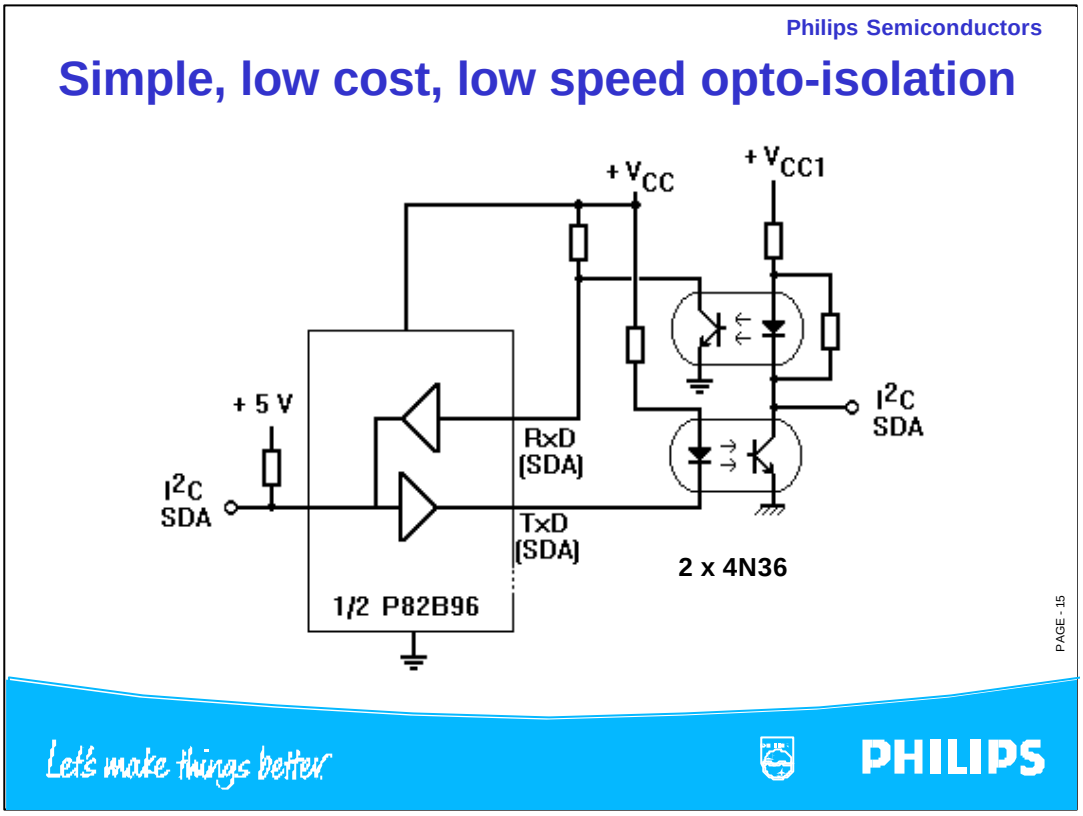
Then the generated SDA 'low' (acknowledge) must travel back along the wiring to the first chip. If the wiring is 100 metres long then the signal travelling time each way will be 500 nsecs. The total travel (propagation) time will be 1 microsecond.

The bus rise-time is allowed to be 1 microsecond, and the fall time 0.3 usecs. Data set-up is 0.25usec. That's already 2.55 usecs total, even if the remote IC acknowledged with zero delay. If the remote chip was PCF8570, and took its maximum 3.4 usecs to acknowledge, then the total delay would be 6 usecs. P82B96 also delays the signals by up to 0.6 usecs.

At 100kHz the nominal clock low is 5 usec, but in practice the bus rise and fall delays change it. The solution is simply to use a SLOWER CLOCK when driving VERY long busses.

The whole point is that the time I2C signals take to travel in long wires must be considered.

Applications: Monitoring groups of gaming machines, factory automation, home automation bus, alarm systems, building automation - lighting or air-conditioning control, access, lifts, monitoring emergency lighting, fire alarms, monitoring groups of food or drink machines.



Galvanic isolation: There have been many attempts at opto-isolation in the literature over the last 10 years.

All result in (and most admit to) problems with glitches resulting from signal propagation delays.

They all block the signal propagation in one direction to prevent latch-up.

So they all produce a 'glitch' during the signal propagation time needed to clear that blocking.

Note that in 82B96 we do NOT block signal propagation in one direction, we modify the logic 'low' level instead.

In this way our 'glitch' is represented by just a transition between our two chosen logic 'low' voltage levels, between <0.4 V and <1V.

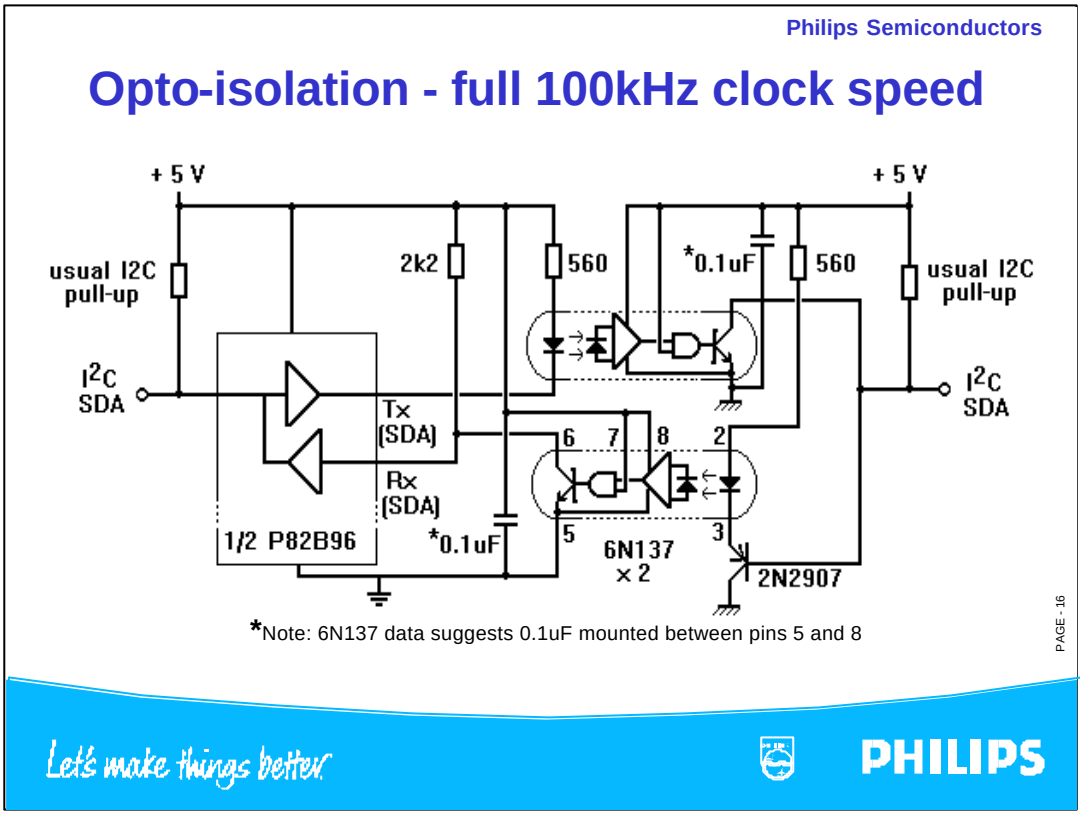
Because BOTH those levels are valid 'low' levels for I2C, we make no real glitch.

This example uses low cost general purpose 4N36 optos with published switching times around 50 usecs for the impedances shown. That will limit the design bus clock speed to around 5kHz. Simple circuits like this can probably be pushed to about 20kHz by changing the type of opto or the circuit values.

Low speed applications:

Driving remote displays, security systems, access control systems, data logging, remote control of appliances or lighting.

Use opto isolation in these systems at any points on the bus where safety isolation or ground potential differences require it.



Opto Isolation at full 100kHz clock speed.

Here the single bus line to the left of the circuit is split by the 82B96 into two separate signals RxD and TxD.

The signal TxD goes low whenever an I2C chip pulls the left hand bus low.

(It does not go low when only the P82B96 is pulling that bus down. The P82B96 can only pull the left hand bus line down to 1V. That is a 'low' for all connected I2C chips, but not low enough to cause TxD to go low. TxD only goes low when the bus is pulled below 0.6 V.)

Current flows in the led of the upper 6N137, causing the right hand bus to be pulled low.

A low on the right hand bus turns on the 2N2907 and the led in the lower 6N137.

In turn, that pulls the RxD input to the P82B96 low.

While there is a low at RxD, the left hand I2C will be clamped so it cannot rise above 1V.

However it is already below 0.4 V (I2C spec) so nothing much happens.

If the I2C chip holding the left hand side low releases that bus then the bus voltage will rise up to 1V, where it becomes clamped by the P82B96 because RxD is low.

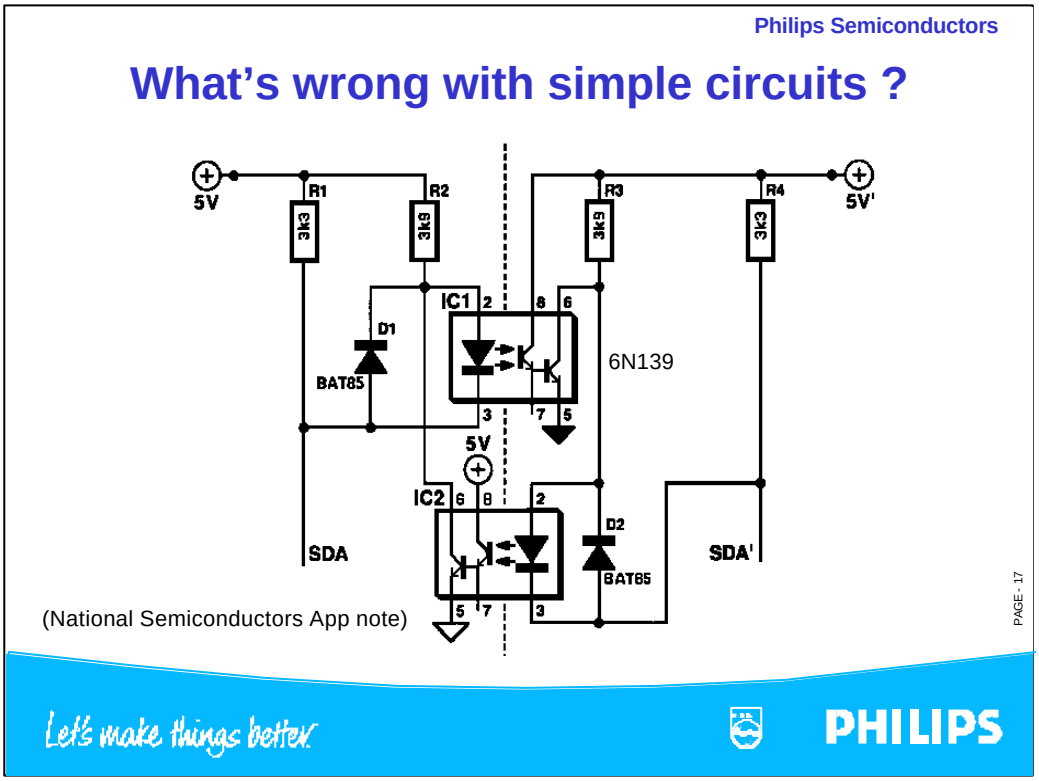
(If RxD happened to also be held low by an I2C chip on the right hand bus then the left hand bus just stays low (at 1V). Note carefully there is NO glitch to a logic 'high' level.)

Once the left hand external I2C chip releases that bus, the P82B96 will release TxD. The upper opto turns off, releasing the right hand I2C and the lower opto drive.

If there is no other chip holding the right hand bus low, then RxD will go high. In turn P82B96 will release the 1V clamp on the left hand I2C bus and it will go high.

There are always propagation delays, we only eliminate the false high logic level on the I2C bus

For no chips on the right hand side, the total 82B96 delay seen on the left hand bus, after release by an I2C chip, will be [delay while left I2C rises from 0V to 0.6V + delay in 82B96 till TxD goes high + delay in P82B96 after RxD goes high until I2C clamp at 1V on left I2C is released + delay while left I2C rises from 1 V to logic high level].



This circuit, from a National Semiconductors Application, is typical of dozens of published circuits for opto-isolating the I2C bus.

There is a simple test to see whether a circuit will generate unwanted glitches on the bus…

Take one side low, take the other side low, release the first side.

In a proper system the first side should just stay low, held by the second side.

Let's do this to the circuit shown.

Take the left side SDA low. The led in opto IC1 is turned on by current in R2.

So the transistor in IC1 turns on. The isolated SDA' is pulled low via D2.

Now take the right hand side SDA' low. It is already low. Not much happens, except D2 no longer conducts.

Release SDA.

There is nothing to hold it low, IC2 isn't on. So it goes high - the beginning of an unwanted 'glitch'.

So now the led in opto IC1 turns off. Its transistor turns off after a delay time.

Now R3 can pull up pin2 of the led in opto IC2 and current flows in it to SDA'.

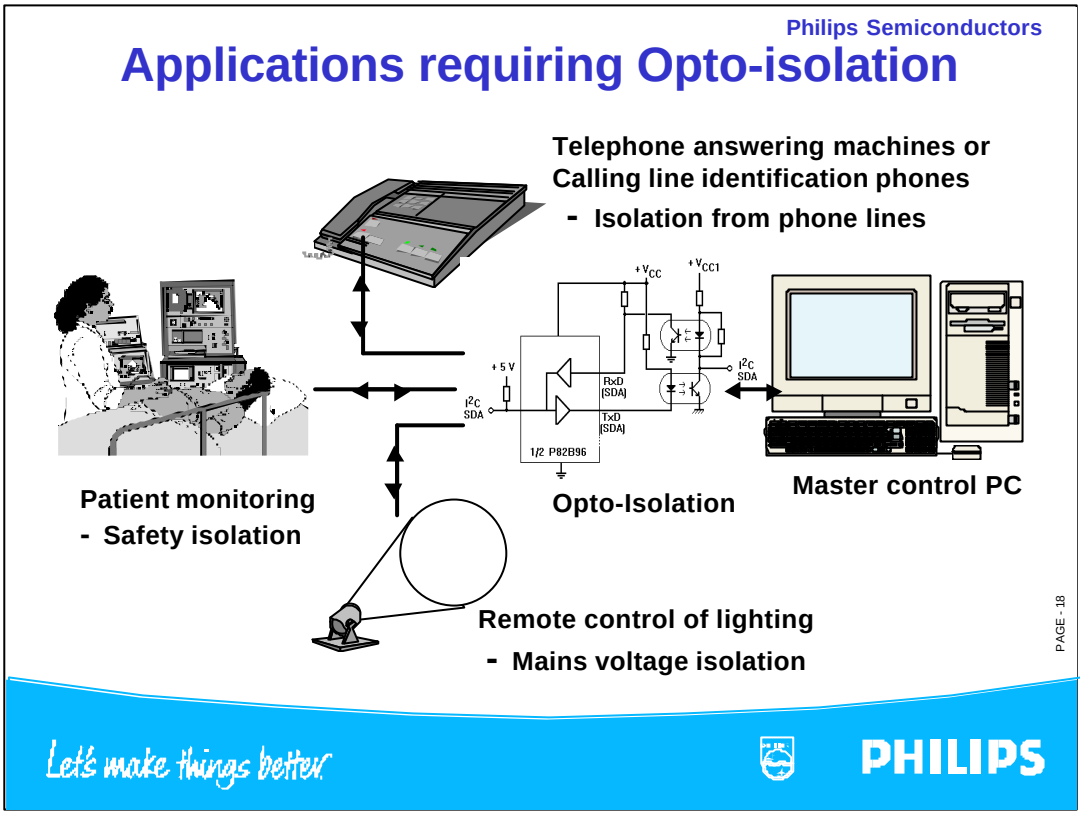
After another delay the transistor in opto IC2 turns on and SDA is pulled down again to the correct state.

But notice SDA was high for the time it took IC2 to turn off, plus the time for IC1 to turn on.

The 82B96 doesn't fail this test.

Another minor point. Note when SDA first went low it had to sink current in both R1 and R2.

The allowable total is 3 mA. Only about 1 mA is available for the opto, so they need a high gain Darlington version (6N139). P82B96 can supply up to 30 mA and work with cheap optos.



Applications requiring isolation of I2C bus.

1) Digital telephone answering machines (Philips PCD6002H), Fax machines, feature phones and security system auto-diallers are connected to the phone line and often powered from the 110/230 V mains via double-insulated 'plug-pack' dc power packs. Many use Microcontrollers (eg. PCD33xx), and some will already have I2C busses. Any other interfaces, eg. connecting to the product's internal I2C bus, will require safety isolation.

Typical phone chips using an I2C bus include DTMF generators (PCD3312C), CLI decoders (Calling line identification receivers eg. Philips PCD3316), and EEproms for memory.

There will be MANY applications requiring CLI data to be recorded - logging callers to sales desks, emergency call look-up of caller location, security dial-up logging, telephone votes etc.

The innovative uses for CLI data will grow like the innovative uses of Internet !

2) Medical equipment requires safety isolation of the patient connections. Any power for the isolated circuitry must be passed via isolating transformers. The data paths are sometimes transformer coupled using carrier tones, but they could also be via opto-isolated I2C.

3) Lamp dimmers and switches can be controlled over I2C data links.

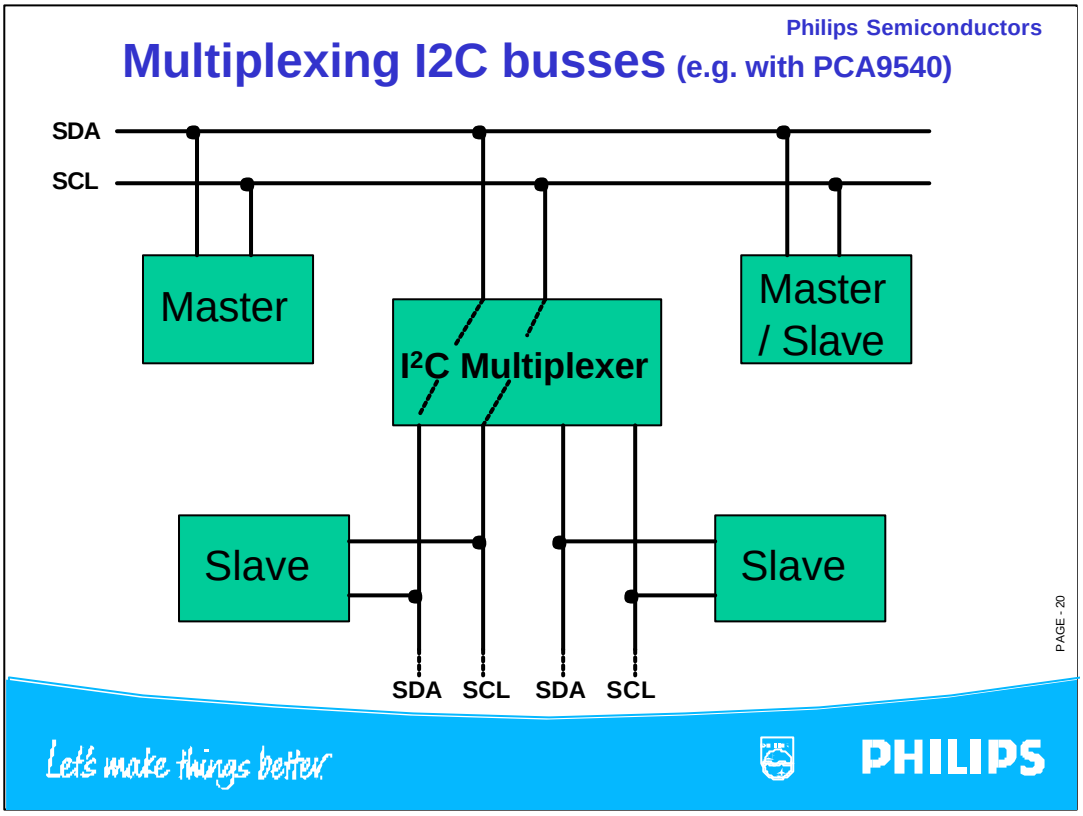
Each light in a disco or live stage production could have its own identity and be individually computer controlled from a control desk or computer via I2C.

Dimming (phase control) can be done with small micros or TCA280B from IES. Putting the phase controller inside each lamp will make it easier to meet EMC rules - lower power wiring radiation.

Because the triac and controller need to be connected to the power mains, any control signal needs to be opto-isolated. I2C control has advantages - each unit can have local 'intelligence' so a command might be "Light 57, dim from full to 10% over 3 seconds starting in 5 secs". Each unit can be "read" to check what was programmed.

Some other ways to 'extend' I²C

- **Multiplexing switches**
  - activate alternative sections of bus in a large system
  - can overcome addressing limitations for multiple devices
- **Voltage level shifting** (without buffering)
  - to mix 3V and 5V logic families
- **Voltage level shifting** (with analog buffering)
  - mix logic families and include long wiring runs
- **Convert to another bus hardware** (CAN, RS485 etc)
- **Dynamic 'sub-addressing' for large systems**



Multiplexers are currently available with 2 and 4 channel outputs.

In the simplest example a multiplexer is addressed and then a control byte is written to the multiplexer to enable a connection from the main input bus to either one (or none) of the outgoing busses.

Philips Semiconductors

Which Buffer ??

Application	PCA9515	PCA9516	P82B96	P82B715
Data sheet supply voltage range Vcc	3.0 – 3.6 V	3.0 – 3.6 V	2 – 15 V	4.5 – 12 V
Nominal logic levels supported (range)	Vcc to 5.5 V	Vcc to 5.5 V	Vcc to 15 V	Equal/less than Vcc
Allows I^2C bus logic level shifting (range)	3.0 – 5.5 V	3.0 – 5.5 V	2 – 15 V	No level shifts
Allows interconnecting I^2C buses, each 400pF	yes, 2	yes, 5	yes	no
On-chip bus sink current capability	I^2C	I^2C	10x I^2C	10x I^2C
Drives lower impedance "I^2C like" buses	no	no	yes	yes
Max (multimaster) bus capacitance supported	800 pF	2000 pF	unlimited	3000 pF approx
Allows inter-working of I^2C and SMBus	yes	yes	yes	no
Designed operating I^2C clock speed	400 kHz	400 kHz	100 kHz	100kHz
Typ.propagation delay (excluding contention)	100 nsec	120 nsec	300 nsec	400 nsec
(Multi-master) bus system configuration	hub/star	hub/star	multi-drop	multi-drop
Splits I^2C to Tx/Rx allowing opto-isolation	no	no	yes	no
Releases all I/O if Vcc supply fails	yes	yes	yes	no
I/Os can be pulled above chip's Vcc level	yes, to 5.5V	yes, to5.5V	yes, to 15 V	no
Logic "buffer enable" input(s)	yes	yes, 4	no	no
Supply current (typ)	2 mA	7 mA	1 mA	16 mA
Packages	TSSOP/SO8	TSOP/SO16	DIL/SO8	DIL/SO8

PAGE - 21

Let's make things better

PHILIPS

The I2C 'repeaters' PCA9515/6 are designed to allow the interconnection of low voltage I2C busses without transferring the capacitive load of one bus to the other bus.

So they can effectively increase the maximum bus capacitance from 400 pF to 800 pF or more.

The impedance (pull-up) of each bus is limited to the traditional range of values.

The use of more than one 'repeater' IC is restricted to systems having the master (or masters) at the 'star' point of all busses because bus signals can never propagate through two 'repeater' ICs.
Its low logic thresholds enable ALL the I2C specs to be met, whereas 82B96 makes some compromises to tolerate slightly more 'noisy' bus conditions.

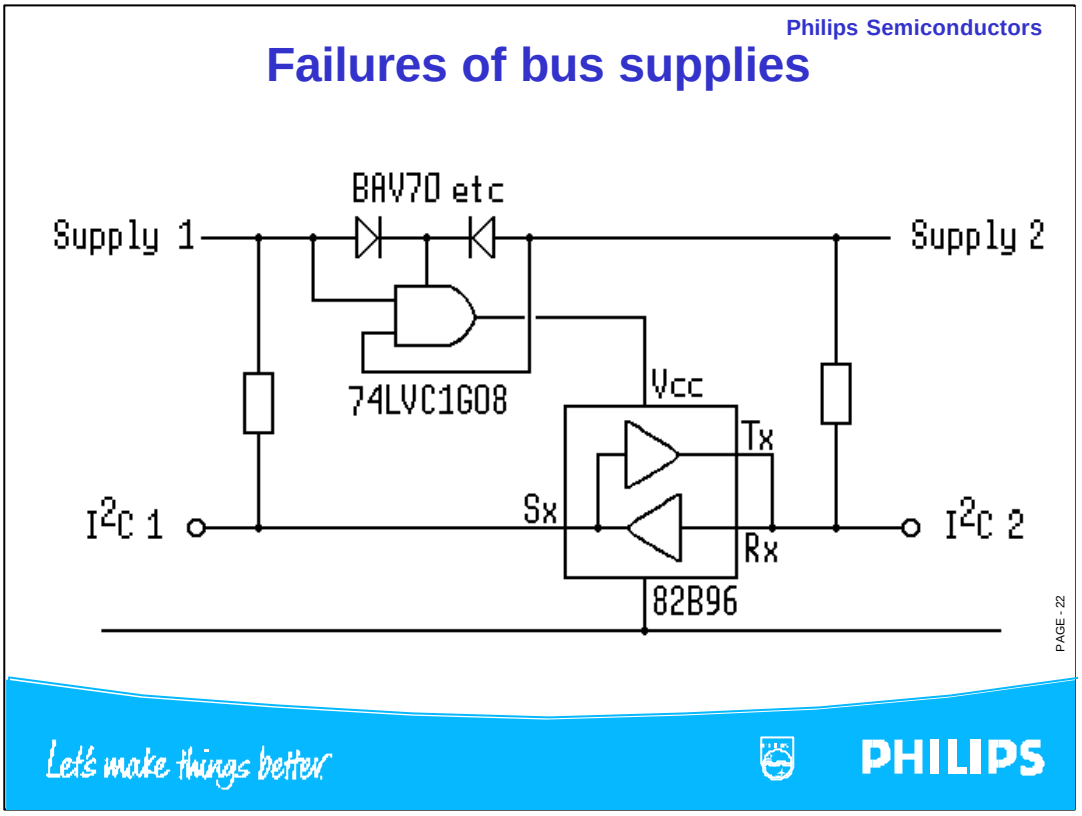
The I2C 'buffer' 82B96 is designed to allow the interconnection of an unlimited number of I2C busses without limitation on the voltage or impedance of those busses.

In its simplest connection it performs the same function as the 'repeater', but multiple connections with multi-master operations are also possible by using the 'buffered' bus side of the chip that has perfectly normal logic voltage levels (and thresholds) and no connection restrictions.

The 'buffer' can also be used to split the bi-directional I2C signals into pairs of uni-directional logic signals making those signals suitable for transmission through opto-couplers or over any re transmission path. The 'buffer' when used in reverse will re-combine unidirectional signals to form I2C signals.

The 82B96 has ten times normal pull-down capability on its 'buffered' bus. So it can directly drive opto-couplers to 30 mA or drive an I2C bus with 10 times lower impedance pull-ups and thus support 10 times greater bus wiring capacitance (4000pF).

When applied with further external amplification (eg. power transistors) there is no theoretical limit to the permitted bus impedance/capacitance. Of course there are practical limits.



ICs like 82B96 and PCA951x are characterised to release all bus lines if their supply 'fails'. But 82B96 is guaranteed to operate correctly with just 2V supply and in fact it operates down to about 1.5V.

Its supply needs to fall below 1.5V (guaranteed value is 1V) before the chip actually releases the busses.

Practical systems are quite likely to have partial failures of their power supplies, called 'brownout', and the system might require bus release under these conditions.

Where it is practical to access two or more logic supply rails, connected to the bus pull-up resistors, the simple and-gate solution shown here can be useful.

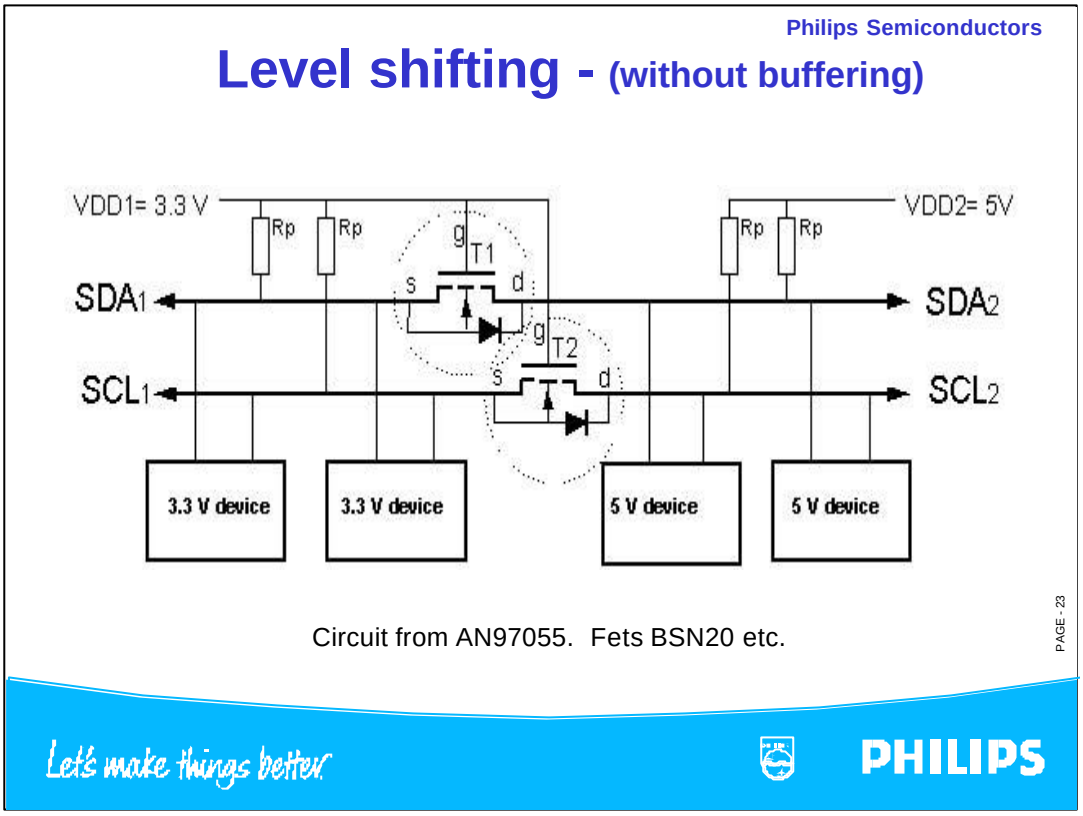
If supply 1 fails (drops below half the level of supply 2) then the Vcc supply to the 82B96 will be removed and the busses will be released.

The BAV70 diodes provide an 'or' power supply to the gate.

It is possible to extend this concept to more accurate supply sensing, for example each supply can be sensed using a 'microcontroller reset' IC and the 'supply valid' output of that chip can be used to control Vcc on the 82B96.

When the supply rails are different, for example 3V and 5V in the example circuit, the higher supply will become the 82B96 supply so its logic threshold at Rx will be half that voltage.

Keep in mind that the bus logic voltages used with 82B96 do not need to be the same as the supply (Vcc) of the chip. All I2C chips pull the bus to nearly 0V/ground. The noise margin for a 'low' is then approximately half of the applied 82B96 supply (Vcc). As long as that is satisfactory, the actual value of Vcc on the chip is not so important.



Simple bus level conversion with Mosfets. See AN97055

MOSFETS like BSN20 can be used to convert I2C signal voltage levels, but they do not 'buffer' the signals. Symmetrical Mosfets like BSS83 (SOT43, 4-lead surface mount) can also be used. They don't have the parasitic diodes shown, and do have gate protection zeners, but are not so rugged.

Any device pulling down its own bus must also sink the pull-up resistor current of the other bus.

So a device at SCL2 must sink whatever current flows in its pull-up resistor (Rp) to Vdd2 on the right-hand side as well as the current in the pull-up resistor (also Rp) of the 3.3 V device on the left-hand side.

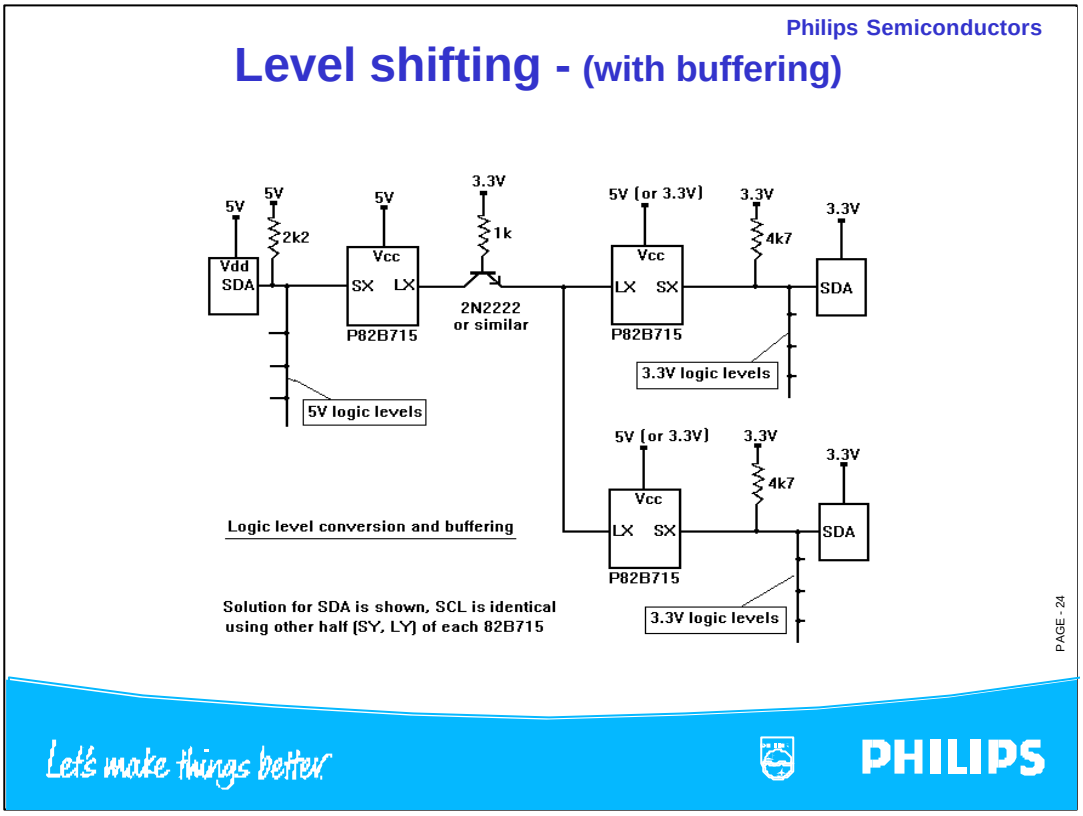
Advantages: Uses no current, physically small - comparable to P82B96 !

Applications: Level shifting in low power systems with just a few chips and low capacitance.

Restrictions: Fets (gates) must be wired on the lower voltage supply side. That is, if used to buffer a 3.3V mobile plug-in device to a 5 V static device then the Fets are fitted into the mobile device where space is usually more restricted.

The Fets must be chosen to have their 'Vgs on' voltage lower than the minimum supply voltage. Typically BSS83 operates to below 3 V supply, and BSN20 has a gate threshold of 1.8V max.

Extension: Failure of the 3.3 V supply will not result in clamping the 5 V bus but the reverse is not true. A similar solution using TWO fets for each bus line can provide full isolation of either failed bus section. But those 4 fets then have 16 pins to connect, and occupy more space than P82B96 in SO8 with its 8 pins.



When level shifting is required, the 'current gain' of 82B715 can be used to simplify the interface circuitry to just one low cost general purpose bipolar transistor such as 2N2222.

When the 3.3V bus goes low, the emitter of the 2N2222 is pulled low and acts as a 'common base' amplifier pulling its base down to about 0.7V and its collector, also LX of 82B715, down to about 100mV above its emitter. That causes a normal 'low' level on the 5V bus.

When the 5V bus goes low the 2N2222 operates in 'inverse' mode with its collector acting as the emitter but still operating as a common base amplifier and pulling its emitter (and LX of 28B715) to a normal bus 'low' level.

The inverse gain (hFE) of general purpose transistors is typically around 10.

In systems approaching the loading limits of the I2C bus it is preferable to avoid special high speed switching transistors like BSX20/2N2369A, because their inverse gains can approach 1, but in practice even those transistors will function ok.

In the circuit of the slide, with just two connected 3.3V busses, the effective loading of an I2C device on the 5V bus is its pull-up to 5V in parallel with the BUFFERED loading equal to the two 3.3V pull-ups and the 1k base resistor in series with one Vbe.

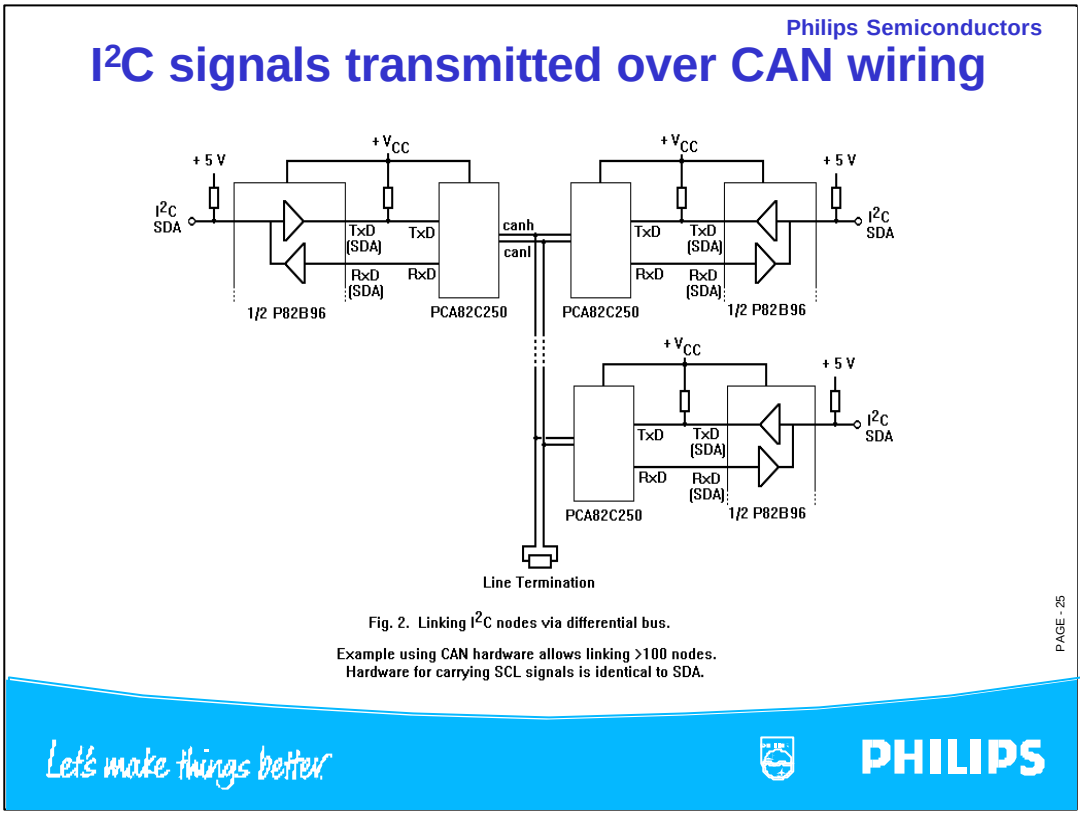
So the current when a 5V device is holding the bus low (at say 0.4V) is typically:

(5-0.4)V / 2k2 + 1/10 [(3.3-0.4)V/4k7 + (3.3-0.4)V/4k7 + ((3.3 - 0.65 - 0.4)V/1k]

= 2.1 mA + 1/10 (3.5mA) = 2.4 mA.

The 1k base resistor is providing 2.2 mA to keep the 2N2222 saturated for the 1.2mA load.

In 'inverse' operation the load is (5-0.4)V/2.2 = 2.1mA, so its requires a gain just less than 1.



Linking I2C nodes via twisted pair busses:

While twisted pair busses are an important market segment, and we show how to use CAN interface ICs to convert to a differential bus, it probably won't be applied unlessthe lower 'power to noise' performance of differential signalling is really needed.

Obviously this is not a cheap hardware solution, but it is a simple, practical illustration of the system possibilities.

For any large market requirement it would be better to include the differential bus driver function in a single, special purpose, interface IC.

The real point is that this conversion is now possible. Previously, bus latch-up precluded it.

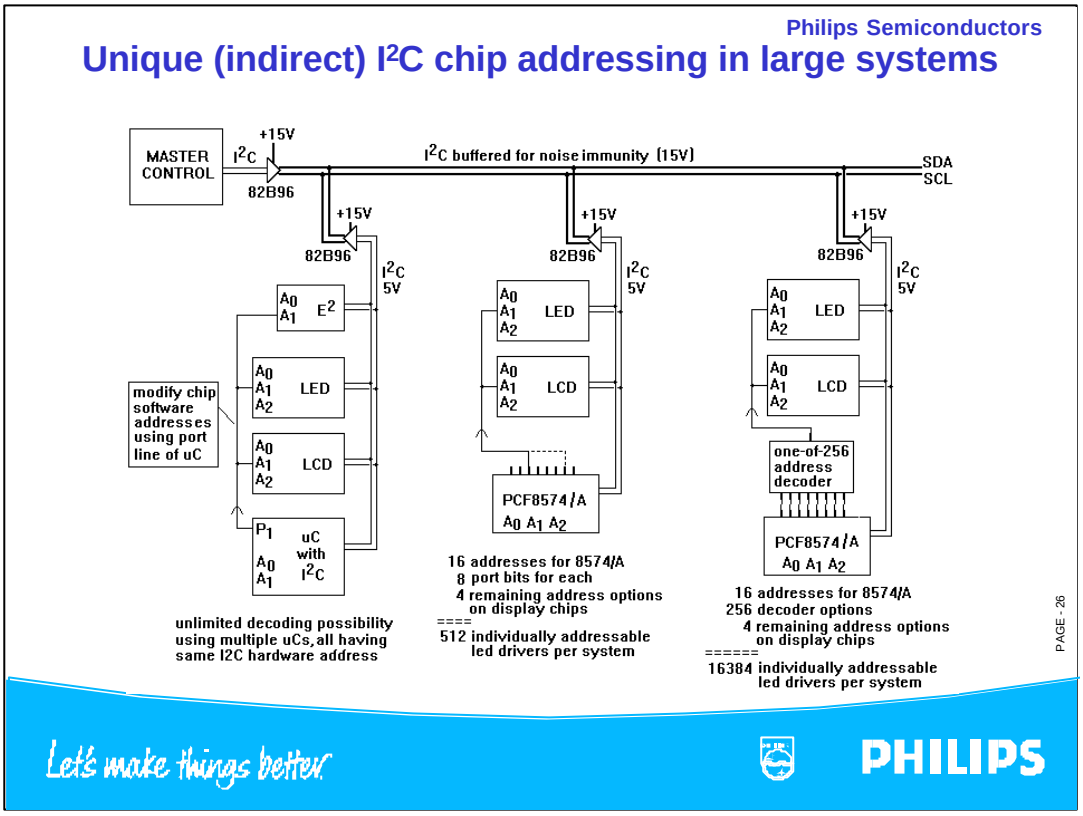
Small-signal differential signalling, with idle state 'power-down' of the bus drivers, will have a lower power consumption than 'brute force' signalling with 12 V I2C logic signals having the corresponding higher currents of lower impedance signalling.

Since P82C250 allows >100 chips on the CAN hardware, we have here an example of how >2000 I2C chips could be used in one system (ie. 20 x I2C chips at each CAN drop-off).

Be careful to note that the CAN bus hardware is only a transport medium for carrying I2C signals.

The I2C signal format is NOT being converted to CAN bus data format by PCA82C250.

It is theoretically possible to use a uC to convert the data format to CAN as well, but that is not illustrated here.



Obviously large numbers of I2C chips in a single system can present addressing problems, but generally it is NOT a problem for Microcontrollers.

As shown on the left, there could be 'unlimited' microcontrollers all sharing one common hardware I2C bus address.

The master controller addresses all micros and then transfers several bytes of information to all the micros.

The micros then interpret that information - extracting the addresses and comparing it to an address stored in EEProm or OTP Rom.

The information can also contain instruction about the actions required – send back an acknowledge, perform checksum and send it back for security, alter port lines which are connected to the hardware address pins of other I2C chips in the system.

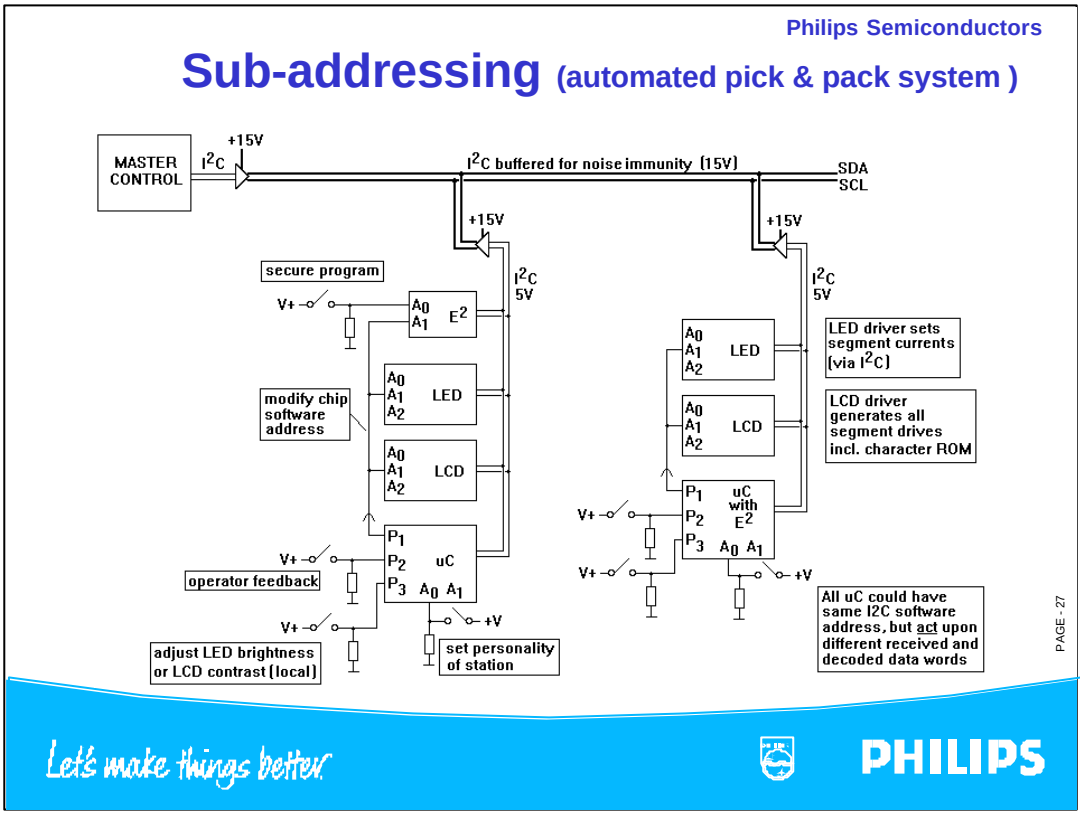
Only the I2C chip whose address is modified, say A1 = high, will be able to be DIRECTLY addressed from the master control.

Notice the data for individual I2C chips does NOT need to be passed via the micro, the master control can load directly into the peripheral chip that has been uniquely selected.

Sub-addressing by means of port expander chips like PCF8574 has been widely used in telephone exchange systems. In this case the A0/A1/A2 pins of other I2C chips are modified by first writing one data word into several PCF8574s which shared one common I2C address.

Centre example: 8 different LED driver chips type SAA1064 can be I2C addressed via their three address select lines. The master will only use LED driver addresses in which A1 = 1.

Then there are still 4 address possibilities via A0 and A2. The A1 pin is only set to 1 by addressing several PCF8574s and loading all their ports with a word that has only one bit high. There are 8 options for this. There are 8 hardware I2C addresses for PCF8574, and another 8 for PCF8574A. The total number of led driver chips that can be uniquely addressed and then directly loaded with data now exceeds 500. This is an illustration, not a suggestion !



Example of an automated warehouse pick-and-pack system.

The Master Control might be a PC with I2C generated in software and converted via a simple logic IC interface on one of its serial ports.

The I2C signals are boosted to 15V levels and the pull-ups reduced in impedance to 500 ohms (distributed around the bus) to enable communication over at least 300 feet.

At each remote station there is a microcontroller.

All microcontrollers have the SAME hardware I2C address on the I2C bus.

Master sends data to ALL uCs. Each uC decodes that data, extracting a unique ADDRESS and some data about actions required.

One action can be to set the address/personality of a station AFTER installation.

Each uC has an address, stored in EEProm. If the address matches then that uC has been selected for action and will process the following data.

Note that the address is **not limited**, it could be 10 x 8-bit bytes if needed.

The addressed uC will output on a port pin a logic signal which MODIFIES a hardware address pin of other standard I2C chips at its station.

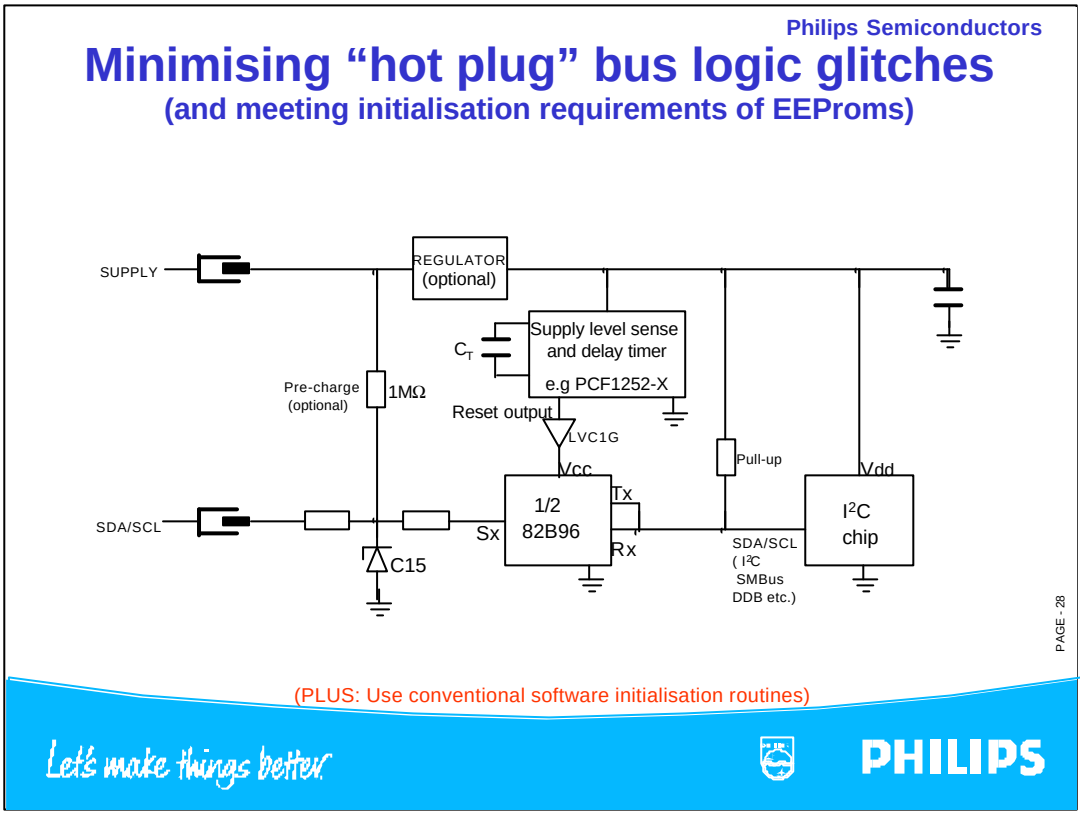
Next the Master Control writes DIRECTLY to those standard I2C chips, loading LED or LCD display information about actions required.

It could also send data to print package labels - anything that's needed.

The master then selects the next station and the current uC de-selects its I2C chips.

When picking is complete the operator can acknowledge via a switch, sensed by the uC on a port line. The uC can locally control LCD contrast or LED brightness over the same I2C bus.

Because I2C is bidirectional, and multi-master, the uC can wait for the bus to be free and send that acknowledge back to the Master Control.



In this example a 'module' containing, for example, an I2C EEProm chip will be 'hot plugged' into equipment with power on and the I2C bus already active.

The hardware requirements are..

1) Cause no disturbance to the main running I2C bus (no glitches to 'low')

2) Ensure the SDA/SCL lines of the EEProm follow the rise of the EEProm supply rail with no 'glitches' to 'low' that could result from activity on the main equipment's I2C bus

3) Provide effective ESD protection for the I2C chips in the module, for safer handling.

Meeting 1) requires that the bus must not be pulled low by, eg., diodes to the supply rail inside the EEProm chip during the time the module supply is being charged from zero to proper operating supply. (-- or even just from charging the stray capacitance of the module bus lines from zero volts).

This is achieved by delaying application of supply to 82B96 until after the EEProm supply is valid.

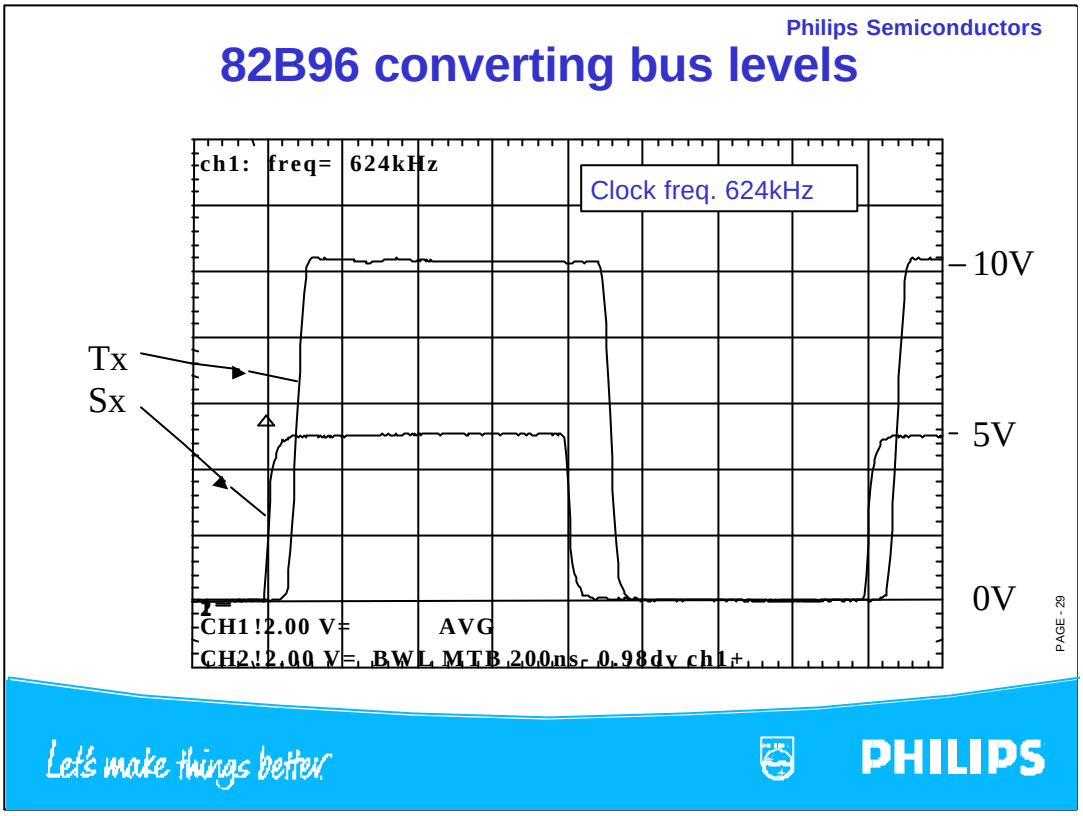
Without supply, the 82B96 will open circuit its I/O lines. At worst the active I2C bus must charge the stray capacitance of the C15 zener (about 30pF) and 82B96 input (about 5pf).

When the supply/ground connections can be arranged to mate before the bus lines, 'precharging' the zener and 82B96 input can prevent even small 'dips' on the main I2C bus, assuming it is high.

Requirement 2) is also met because 82B96 releases Rx/Tx so the pull-ups on the EEProm will cause the SDA/SCL lines to follow the rising supply.

The 15V tolerant I/Os of 82B96 allow ESD protection using high voltage zeners having much lower capacitance than low voltage ones. (In any case 5V chips cannot be protected by 5V zeners). Small series resistors in the bus lines provide additional reduction of 'glitches' on the active bus.

Because PCF1252 may 'enable' the bus connection during an I2C transmission, it is still necessary to use any software initialisation routines as recommended by the EEProm (etc.) chip supplier.



Scope waveforms show 5V I2C signals (Sx) being converted to just over 10 V levels at Tx (Ch 2).

These signals show the transfer delays between the I2C bus side and Tx output.

In this picture Tx has NOT been linked to Rx, so that the simple one-way transfer can be seen clearly.

In order to show the propagation delays clearly, a clock frequency of 624kHz was used.

At normal clock frequencies this delay is negligible.

There is no external capacitive loading on either bus.

The horizontal axis is 200 nanosecs/division.

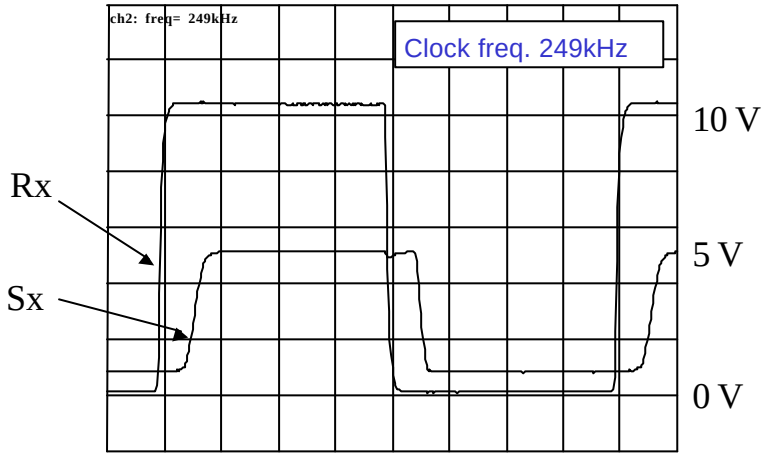
Note that a 'high' level is recognised for Sx (Ch 1) at about 0.6V.

Data sheet specification of the delay when the busses are rising is specified between the time Ch 1 reaches this 'high' and the time when Tx (Ch 2) reaches half the Vcc level. In this case Vcc is just over 10 V so the Ch 2 'high' level will be about 5 V.

When the I2C bus is falling, the delay is specified between the time when the I2C bus reaches the low level of 0.6 V and the time when Tx (Ch 2) has fallen to half Vcc.

Philips Semiconductors

82B96 Rx signal drives Sx



Horizontal: 500nsec/div. Vertical: 2 V /div.

PAGE - 30

Let's make things better

PHILIPS

Waveforms show an input driving signal at Rx coming from a 10 V bus and the the resulting output at Sx with a pull-up to a 5 V supply.

Note that when Rx goes low the Sx is pulled down to about 0.8 V.

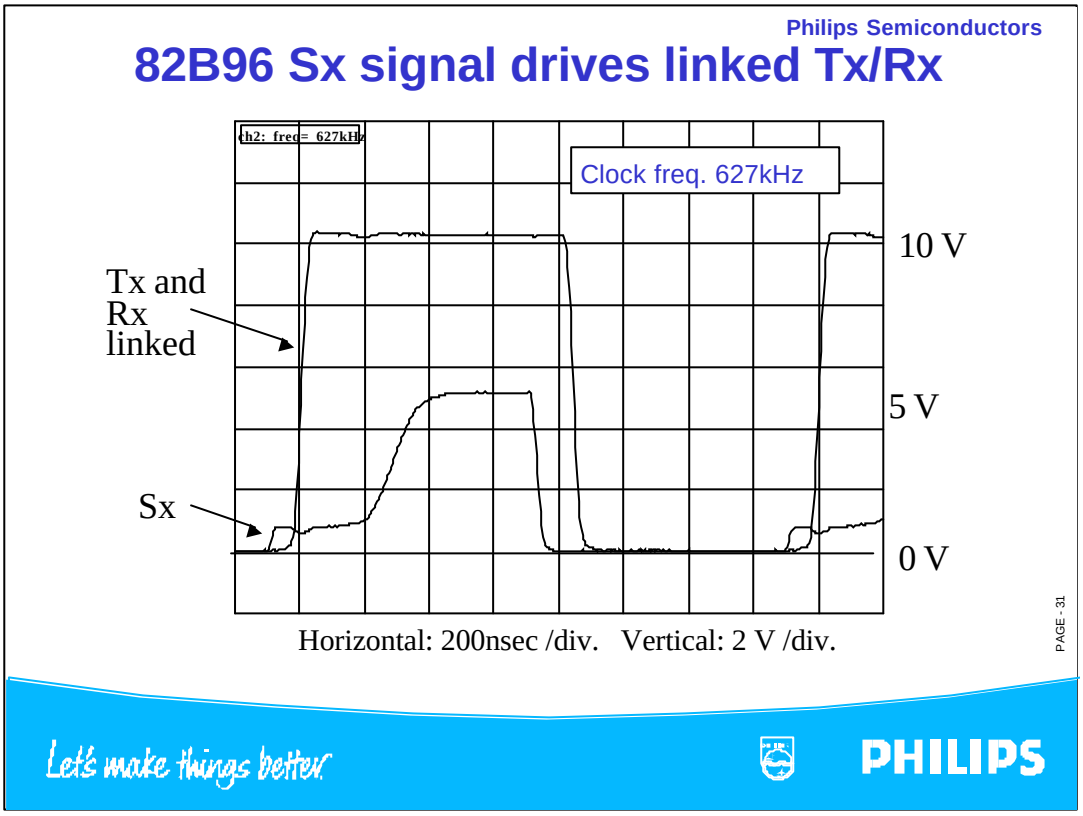
That's a 'low' for the I2C bus - a 'low' is defined at 1.5V.

But the Sx isn't low enough to be recognised by P82B96 as an external chip pulling the bus low. (External chips will pull the bus below 0.4 V)

So the 'low' on Sx does NOT cause Tx to go low.

Note that a clock frequency of 250kHz was used to exaggerate the propagation delays.

On both rising and falling edges, the delay Rx low to Sx low is specified at 300 nsecs.



In this slide we use a high clock frequency (627kHz) to exaggerate the delays that occur when a signal at Sx drives a new "I2C" bus by linking Tx and Rx.

On the extreme left an external I2C device has been holding Sx low.

That has caused Tx to go low, and also Rx because it is linked.

When the I2C chip releases Sx, the Sx is only free to rise to about 0.8 V, where it becomes clamped because Rx is low.

But Sx is now above the low threshold (0.6V) at Sx, so after about 100nsec delay the Tx goes high.

That means Rx goes high.

After the specified 300nsec propagation delay the Sx is released and rises to 5 V.

The total delay, from release of Sx by an external chip until Sx actually goes high is about 400nsec.

Notice that Tx actually goes high BEFORE Sx goes high.

The release of the I2C bus is transmitted to the Tx/Rx bus with only 100nsec delay, even though the logic voltage at Sx remained 'low' for 400 nsecs.

When Sx goes low again, Tx will go low after the specified 100 nsec delay.

Remember these waveforms are greatly exaggerated by the use of 627kHz drive waveforms.

When operated at 100kHz clock these delays are negligible, unless very long bus wiring adds such large delays that the total delay approaches 5usec.

(The signal delay caused by cables will be about 5 nsec/metre).