I²C as a low-cost Field Bus in the Combustion Laboratory?

Gunnar Arndt
vacos GmbH, Nagold, Germany

Abstract

An inexpensive but practical laboratory setup was previously developed to test the use of gas sensors for early fire detection in buses [1]. The improved and extended setup is presented in another paper [2]. The current paper specifically reports the results of different electronic components used in the setup. The components include Linux-based single-board computers, various sensors connected by the I²C bus, and devices to extend the bus's range. Although I²C is standardized and well established, interoperability was a significant issue. This report does not focus on technical details, but on practical aspects. It is mainly intended to aid in the selection and combination of components that seem useful for combustion research by their specification, while allowing identified issues to be avoided. Most of those surprisingly occur in regular use of I²C as an on-board bus, not specifically in relation with range extension, for which reason the insights presented in this paper may come in useful for the former use case as well.

Keywords: I²C, field bus, sensor, multiplexer, single board computer

Introduction and overview

In recent years, companies like Adafruit, Sparkfun, and MikroElektronika have made sensors and other components on breakout boards available. Although these are intended for hobbyists, they are also useful for initial evaluation, or for test setups for the laboratory or for field tests that are not intended for mass production. Many of the devices are connected by I²C. SPI is also often available but is not considered here as it is less standardized than I²C, and requires more wiring. I²C is originally specified as an on-board bus to interconnect chips, i.e. for short distances [3]. For longer distances, various field busses like CAN, Modbus, and PROFIBUS are available, but are overall more costly to install, or often require additional technical effort to connect certain devices. When using a field bus, each sensor location requires a separate microcontroller, which translates between the local on-board I²C bus and the field bus, thus requiring more complex hardware and software. Readily available and widely used microcontroller boards like the Arduino range are available for that purpose, but are beyond this paper's scope. As there are established ways to extend the range of I²C, and as there are reports of success from other companies, in use for e.g. plasma cutting machines, it appears reasonable to avoid additional microcontrollers just to translate between different data buses. Surprisingly, bus range was not found to be an issue in the considered installations. Compatibility issues were more relevant, but could be addressed, and some of the tested devices were even used for data collection in a test vehicle. Wire lengths of up to 10 m worked well with suitable line drivers or buffers. Nonetheless, the I²C bus is of course unsuitable for actual long-distance alarming or similar applications, which are thus beyond this paper's scope. The presented results are applicable for research work like the mentioned laboratory setup, or on-board implementation of the bus within its specification.

I²C bus basics

A few technical details, which should be considered when using I²C over long lines, and possible problems that were identified in the laboratory setup are briefly covered. I²C consists of a clock line (SCL) and a data line (SDA). It is a multi-master bus, i.e. each device connected to it may initiate a communication cycle; the respective master creates the clock signal. In the current setup, only the host device acts as a master, whereas all peripheral devices are so-called slaves. Even in such a single-master configuration, a feature called *clock stretching* is relevant: a slave device can extend a clock cycle in order to make the master device wait for data. Both these features – multi-master and clock stretching – mean that there is no clear direction of data transmission (or current flow, technically) during a given communication cycle. That is a challenge when constructing buffers with the intention to support longer lines. The I²C bus typically operates at a frequency of 100 kHz; more is possible with a reduced selection of devices. A reduced frequency may help to support longer lines, but there are devices that require 10 kHz as a minimum frequency in their data sheet, e.g. the MCP9600 thermocouple measurement converter [4].

Host devices (single-board computers, SBC)

The project, during which the presented insights were acquired, required data to be collected both in a combustion lab, and in test vehicles. As the latter use case also required telematics, i.e. the collected data to be transmitted to a data center, relatively powerful ARM computers running Linux as an operating system were used throughout the project. The *Sitec S4* served as a telematics device. Although there were some glitches in the used samples, which e.g. caused the real-time clock and Wi-Fi to fail under different conditions, it generally showed good stability and hardware compatibility with I²C, and it has an integrated P82B96 buffer for long lines. The well-known *Raspberry Pi 3* ("Raspi") was used as a host device in the laboratory for an extended period of time; it required additional cooling in summer and revealed compatibility issues only during the upgrade of the laboratory setup when I²C devices, namely the MCP9600, that rely on the I²C clock stretching feature were integrated. It is known that the Raspi's CPU has a bug that prevents the reliable use of clock stretching. According to tests performed in this project, the often recommended workaround of reducing the bus's clock frequency had no beneficial effect; even the use of software emulated I²C via GPIO pins could not solve the problems [5]. Possible replacements for the Raspi were explored. Texas Instrument's *BeagleBone Black*, besides the fact that it is notably slower, was found to have the same issues, and was not considered further. Even HardKernel's *Odroid C2*, which is being used as a host device in the lab, does not reliably support clock stretching, but could at least be tamed by reducing the bus's clock frequency. HardKernel's guideline [6] suggests an absolute value, but setting the minimum of 10 kHz, which all used slave devices support, resulted in an actual bus frequency of 6.25 kHz. The configuration value was increased until a clock frequency above 10 kHz was achieved.

I²C bus range extenders, isolator, multiplexer

As the I²C bus's intended use is as an on-board bus to connect components over a few centimeters of wire length, it operates outside its original specification when being applied as a field bus. For that purpose, however, multiple chips have been available for a long time. The most common one seems to be the buffer *P82B96*, which usually works in pairs in order to encapsulate a long transmission line between two boards. It also has the ability to shift between different bus voltages, but problems occur with multiple concatenated voltage level shifters, as in the case of the two considered breakout boards with the VOC sensor SGP30, which operates at only 1.8 V. Since the P82B96 has two different "sides" – an on-board one, and a buffered line one – combining it with a multiplexer like the TCA9548A requires special care, because the on-board sides must not be facing each other even with the multiplexer in between. That is because it uses a voltage offset within the I²C's limits on the buffered side in order to detect the direction of transmission. The technically less advanced alternative *P82B715* does not require the aforementioned offset, as it relies on current sensing, and offers much better compatibility [7]. It does not offer voltage level shifting, though. It can be combined with the multiplexer TCA9548A, which has at least limited level shifting abilities [8]. An additional disadvantage of the P82B715 is that it is only available for an operating temperature of up to 85°C, which may become a problem in fire detection or automotive applications, whereas there are variants for 125°C of the P82B96. If electrical isolation between bus sections is required, possibly including ground, *ISO1540* is useful [9].

It must be noted however, that it is limited to a tenfold lower bus capacitance on one side than allowable for an I²C bus. This may thus require an additional buffer like the P82B715 to drive long lines. Further buffers can be found in NXP's I2C-bus Components Selection Guide [10].

Environmental sensors and other devices

An analog-digital converter was required for various transducers; the *ADS1115* worked flawlessly in any bus configuration [11]. Besides the gas sensors evaluated in the laboratory setup, several fully integrated sensors for various physical quantities served as a source of reference data. The ones with highest integration, namely *BME280* (temperature, humidity, pressure) and *BME680* (VOC concentration in addition), and the *HYT939* temperature and humidity sensor, all worked well with I²C communication. There were issues with Sensirion's range of sensors. Here the manufacturer only recommends I²C communication for short distances, and otherwise more suitable interfaces like UART. All of them seem to offer CRC32 checksums, which allow a reliable detection of errors during transmission. These errors were, however, not the problem; instead, the bus would latch in many tests after a while. The CO_2 sensor *SCD30* requires clock stretching and was therefore not operable with the Raspberry; it works in combination with Odroid C2, TCA9548A, and – if required for long lines – P82B715. The VOC sensor *SGP30* was new when this investigation was conducted, and was available on two different breakout boards by Adafruit and SOS electronics ("BOB II"). Both include a voltage regulator and voltage level transition for the bus lines, since the sensor itself needs a supply of only 1.8 V. In any tested configuration, the bus latched at an unpredictable voltage level when the SGP30 was addressed – possibly after seconds, possibly after hours of stable operation. Such events are difficult to catch with an oscilloscope, as shown in in Fig.1, as there is no clear trigger condition. Thus, the SGP30 was not considered further, although its specification is promising.

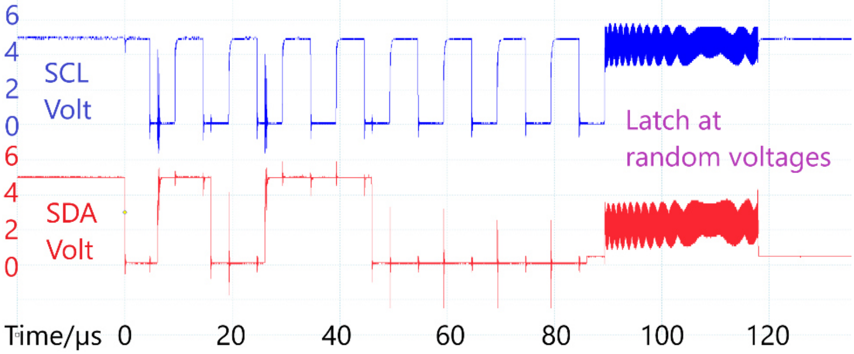

Fig.1. S4's 5V I²C bus latched by SGP30, as observed by a scope.

The particulate matter sensor *SPS30* requires a power supply of 5.0 V, whereas most other sensors are satisfied with 3.3 V, which is relevant because the considered host devices' IO voltage is limited to 3.3 V; voltage level shifting is thus essential. SPS30 runs smoothly at a reduced bus clock frequency when P82B715 acts as a line driver and TCA9548A as a voltage level shifter. In order to measure the temperature inside the laboratory muffle furnace, type K thermocouples were used; initially, *AD8495* was used as a transducer in combination with ADS1115, but it was not very precise. *MCP9600* offers far superior precision, which made its integration vital; as it requires clock stretching, it is incompatible with the Raspberry. An Odroid C2 with clock frequency reduced to less than 25 kHz worked reliably. One issue remains: The clock stretching, which the MCP9600 performs after each byte sent to the master, may still fail, in which case it repeats the previous byte instead of sending the next one. This happens more often at higher clock frequencies and is detectable by reading all three temperature values (hot junction, difference, cold junction) from the device in a sequence, and calculating whether the difference is correct.

Conclusion

I²C is suitable as a field bus in the laboratory environment if carefully integrated, i.e. components need to be matched carefully and the clock frequency should be lowered at least on longer lines. The bus topology should not be too complex, as the combination of line drivers/buffers and multiplexers can lead to problems because of I²C's indeterminate direction of transmission. Checksums, if available, should be considered. Clock stretching and different supply voltages remain an issue; the compatibility matrix in Table 1 may be helpful to work around them. Even a simple oscilloscope is a useful diagnostic tool. Overall, a proper field bus is preferential. Especially beyond laboratory conditions, e.g. in case of an actual alarming application, I²C should strictly be limited to its intended purpose as an on-board bus. The presented results may still be useful for the selection of components.

Table 1. Compatibility matrix. Symbols: + works, ° works at reduced bus clock frequency, - fails, ? not tested.

	Sitec S4	Raspberry Pi 3	BeagleBone Black	Odroid C2	P82B96	P82B715	TCA9548A
P82B96	+	+	+	+	+	+	+
P82B715	+	+	+	+	+	+	+
ISO1540	+	+	?	?	+	+	+
TCA9548A	+	+	+	+	+	+	+
ADS1115	+	+	+	+	+	+	+
MAX31790	+	+	?	+	+	?	+
ISL22313	+	+	?	+	+	?	+
MCP9600	+	-	-	°	+	+	+
SCD30	?	-	-	°	-	+	+
SGP30(1)	(+)	(+)	(+)	(+)	(+)	(+)	(+)
SPS30	-	-	°	°	-	+	+
CCS811	?	-	-	°	?	+	+
BME280	+	+	+	+	+	+	+
BME680	+	+	+	+	+	+	+
HYT939	+	+	+	+	+	+	+

(1) May work for a limited time, but inevitably latches the bus in the given implementations (Adafruit, SOS electronics).

References

[1] G. Arndt, S. Suchy, D. Vialkowitsch, A. Staerz, A. Oprea, U. Weimar and W. Rosenstiel, "Multi-criteria/multi-sensor early fire detection in the engine compartment of road vehicles: Evaluation process of gas sensors," in *16th International Conference on Automatic Fire Detection (AUBE '17)*, College Park, Maryland, USA, 2017.

[2] G. Arndt, A. Staerz, I. Malisi and U. Weimar, "Multi-criteria/multi-sensor early fire detection in the engine compartment of road vehicles: Gas sensors' performance at elevated ambient temperature," in *17th International Conference on Automatic Fire Detection (AUBE '21)*, Mülheim an der Ruhr, 2021.

[3] NXP, "UM10204: I2C-bus specification and user manual," 04 04 2014. [Online]. Available: https://www.nxp.com/docs/en/user-guide/UM10204.pdf. [Accessed 18 10 2019].

[4] Microchip Technology Inc., "MCP9600/L00 Thermocouple EMF to Temperature Converter, Data Sheet Revision D," 08 2018. [Online]. Available: http://ww1.microchip.com/downloads/en/DeviceDoc/MCP9600-Data-Sheet-DS20005426D.pdf. [Accessed 21 10 2019].

[5] Premier Farnell Ltd., "I2C Clock Stretching feature in Raspberry Pi 3," 14 03 2016. [Online]. Available: https://www.element14.com/community/thread/49698/l/i2c-clock-stretching-feature-in-raspberry-pi-3?displayFullThread=true. [Accessed 21 10 2019].

[6] HardKernel, "Odroid Wiki: I2C," 08 05 2019. [Online]. Available: https://wiki.odroid.com/odroid-c2/application_note/gpio/i2c#how_to_change_the_i2c_clock_speed. [Accessed 15 04 2020].

[7] Texas Instruments, "P82B715 I2C Bus Extender," 20 2016. [Online]. Available: http://www.ti.com/lit/ds/symlink/p82b715.pdf. [Accessed 21 10 2019].

[8] Texas Instruments, "TCA9548A Low-Voltage 8-Channel I²C Switch with Reset," 10 2016. [Online]. Available: http://www.ti.com/lit/ds/symlink/tca9548a.pdf. [Accessed 21 10 2019].

[9] Texas Instruments, "ISO154x Low-PowerBidirectional I²C Isolators," 04 2019. [Online]. Available: http://www.ti.com/lit/ds/symlink/iso1540.pdf. [Accessed 21 10 2019].

[10] NXP, "I²C-bus Components Selection Guide," 05 2013. [Online]. Available: https://www.mouser.com/datasheet/2/302/75017443-260510.pdf. [Accessed 21 10 2019].

[11] Texas Instruments, "ADS111x Ultra-Small, Low-Power, I²C-Compatible, 860-SPS, 16-Bit ADCs With Internal Reference, Oscillator, and Programmable Comparator," 01 2018. [Online]. Available: http://www.ti.com/lit/ds/symlink/ads1113.pdf. [Accessed 21 10 2019].