SDI-12 - A protocol dinosaur on the way to Ultra-low-power IoT

LTE-M, Bluetooth 5 and SDI-12 as an ideal triad

Picture 1: A real-world IoT project for rock monitoring in the Valais mountains (Switzerland) – LTE-M coverage is very good, but the place is almost impossible to reach by foot...

The term "fieldbus" is usually associated with established systems from the industrial production area, such as Modbus, CAN or other (often RS485-based) systems.

However, the established systems are often not the best choice, especially for the Internet of Things (IoT) due to the complexity and energy consumption. IoT devices are often powered by batteries and used outdoors and to measure environmental parameters. Usually only sensors are measured, actuators often only play a subordinate role in battery-powered IoT and are often only connected directly to the IoT device (e.g. a flashing light for alarm states). It is precisely in this area that the SDI-12[1] bus can score!

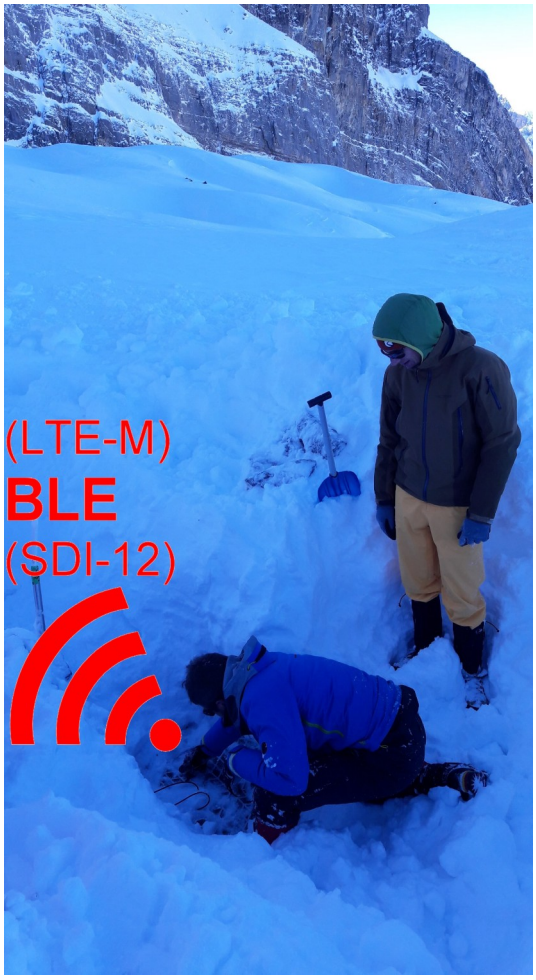
The lowest possible power consumption and maximum uncomplicated maintenance, commissioning and interchangeability of the sensors are the decisive factors for the construction of reliable and low-maintenance systems. You learn to appreciate these factors very much, especially under extreme conditions or in hard-to-reach places! The two photos come from an IoT project, which is supported by the author.

Some typical examples of optimal areas of application for SDI-12 are:

- Natural hazard monitoring (slope landslide, rock stability, …)
- Weather stations
- Water level measuring stations for surface water or groundwater
- Agricultural sensors (soil moisture in the field)
- Settlement measurements on bridges
- Monitoring of pest infestation
- Leak detection
- Containers and garbage levels
- ...

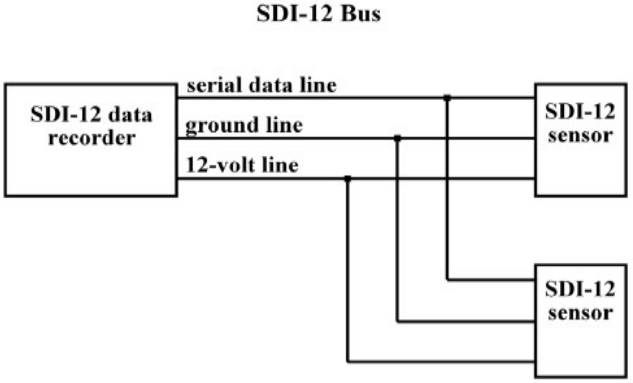
SDI-12 was developed in 1988 by a US agency in cooperation with some private companies and can be used freely. It only requires three lines (supply, signal, GND) and up to 62 sensors can be addressed, whereby each sensor can supply several values at the same time (e.g. pressure and temperature). Since the baud rate is quite low (only 1200 baud), there are no reflection or termination problems, as is the case with RS485-based systems.

Up to> 500 mtr. Cable "may" be laid in almost any topology. The supply is traditionally approx. 9.6V - 14V and the signal line communicates with approx. 0V - 5V levels. The communication is completely in a human-readable format, it is even explicitly requested according to the specifications of SDI-12 that humans can communicate directly with the sensors! Even individual, sensor-specific commands are provided in the specs!



Picture 2: Fortunately, the devices have Bluetooth, otherwise they would probably not be found in the snow ... (at -20 ° C at 3000 mtr.)

Nevertheless (since version SDI-12 V1.3) there is also the possibility of transferring the data with a CRC16 checksum without errors.



Picture 3: SDI-12 bus with 2 sensors (source SDI-12.org)

But one of the most important advantages of SDI-12 (which is hardly found in any other fieldbus) is that sensors are allowed to sleep as needed! A typical example of this is a rain counter: it either wakes up briefly when a rain event has been detected or when its count is queried via SDI-12. In the breaks in between, he ideally consumes “nothing”.

To wake up, the SDI-12 knows a fixed timing, which is a maximum of 100 msec. take. Roughly speaking, a '<Break>' signal is transmitted on the signal line, followed by the address of the relevant sensor ('0' - '9' or a letter) or the broadcast address '?'. All "other" sensors can go back to sleep immediately. This is followed by the command to the sensor in text format, terminated by a '!' Character. The response of the sensor is also text, at the end with one (or two) '<CR> <LF>'. Measured values are transmitted as standard numerical values floating point or integer numbers).

It couldn't be easier! And in the event of problems or for diagnosis, you can read the entire data traffic at any time or manually send commands (such as for setting parameters).

For communication with SDI-12 sensors, a simple terminal program and an RS232 interface (as almost every PC still has) can be used in combination with very few discrete components. A suitable terminal program ("SDI12Term.exe") is in the source code and compiled on the author's Github page.

Here are a few examples (each command is preceded by a '<BREAK>' signal to wake up):

- '? I!' - Identifies a sensor (using the broadcast address '?'). The answer is for example: '513STS AG 4900001.51157252 <CR> <LF>' (The sensor has the address '5' and comes from STS-AG, the rest is unimportant). The only problem with this may be that if there are multiple sensors, they all respond, resulting in garbage ...

- ‘5I!' - The same command, here only the sensor with the address '5' answers.

- '5 M!' - Starts a measurement on sensor '5'. The answer would be e.g .:
  '50012 <CR> <LF>' (sensor '5' has 2 measured values in 001 seconds)

- '5D0!' - Asks about the result. The answer would be e.g .:
  '5 + 0.00180 + 26.15 <CR> <LF>' ('5' is the address, the values are 0.0018 bar and 26.15 ° C, the units result from the (coded) sensor type from above ('? I!'))

Of course, there are more commands, but essentially only the commands that are actually needed have to be implemented.

There are now hundreds of manufacturers of sensors using the SDI-12 protocol. Most sensors support a maximum of SDI-12 V1.3. The very current version V1.4 is rarely used (they are completely contained in the specifications in chapters 5 and 6. Tip of the author: just ignore these two chapters of the specs).

SDI-12 also allows a sensor to be given its own commands! For this purpose, the 'X' command can be extended, for example for calibration parameters or control of actuators. It is also sufficient to provide a sensor with a minimum number of commands. That makes the whole thing very flexible! Of course you notice the SDI-12 bus grown structure, but that also makes him very likeable from the author's point of view!

Individual sensors are quite simple and easy to build. Therefore the author has published two open source projects "Open-SDI12-Blue" (hardware) and "SDI12Term" (terminal for Windows) [2] [3] (links at the end).

The gateway to the Internet / data recorder

Or simply called "data logger", collects the data from the sensors using the above and other (error-proof) commands and transfers them (otherwise it would not be IoT) directly to an Internet server, often for years and with tiny batteries. In the project above, data loggers with LTE-M (and 2G as optional fallback) for the Internet, Bluetooth for local communication with smartphones and SDI-12 for the sensors were used.

These devices are complex and less recommended for self-construction and there are a lot of high-quality manufacturers [4]. In the mentioned project these data loggers have been used:

Picture 4: A rugged and compact Data Logger, also suitable for other applications than Ground Water. (Source: TerraTransfer)

The 2 ways to the ultra-low-power IoT

In 1988 the concept of SDI-12 with its approx. 9.6V - 14V supply and approx. 0V - 5V signal level was absolutely state of the art. Due to the concept of sleeping, low-power has been part of SDI-12 for a long time.

But SDI-12 has always been implemented quite relaxed. Many sensors can also handle other voltages and often also with lower signal levels. Some manufacturers even advertise supply voltages (and thus also signal levels) from 3.6V upwards and quiescent currents in the µA range!

And these sensors can confidently be called ultra-low-power.

Open-SDI12-Blue - an open source project

95% of an SDI-12 sensor are almost always identical! The author has therefore published the Open-SDI12-Blue project, in which various sensors are implemented on SDI-12:

- various pressure sensors (piezo, ceramic, barometric, …)
- Counter for frequencies and events (rainfall, wind speeds)
- Precise analog value acquisition with high-resolution A / D converter
- Temperature / humidity sensors

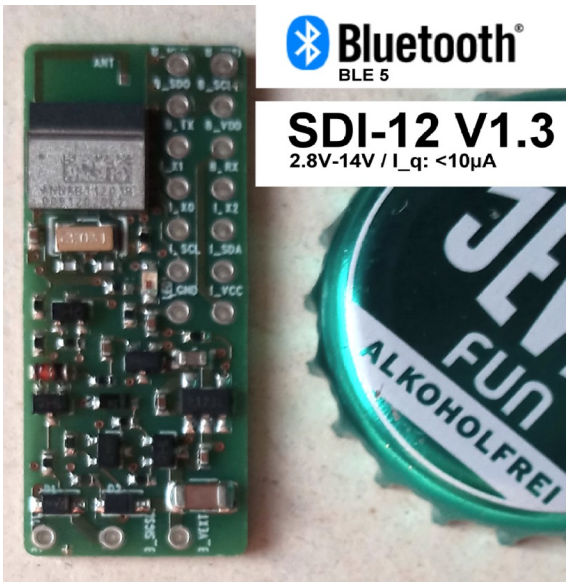
The project is fully documented (including circuit diagrams and software) and is also used in practice.

> ***Important:*** *The commercial use of Bluetooth (including the Bluetooth logos)* ***must be licensed*** *with the Bluetooth SIG[5]! Also please regard the Copyright of the Open-SDI12-Blue-Project (CC BY-NC-ND 4.0, non-commercial use only).*
> *If you plan to use Open-SDI12-Blue for a commercial project, please contact us [7]!*
> *The commercial version of the Open-SDI12-Blue also has a user-friendly secure firmware downloader (via Bluetooth) and access protection (product security) as well as a lot of additinal drivers for other sensors!*

You can easily adapt your own sensors yourself. There are several port pins available that can be programmed very flexibly and easily, for example as an I2C bus, interrupt inputs or as a simple analog input. An SoC from the nRF52 series is used as the CPU:

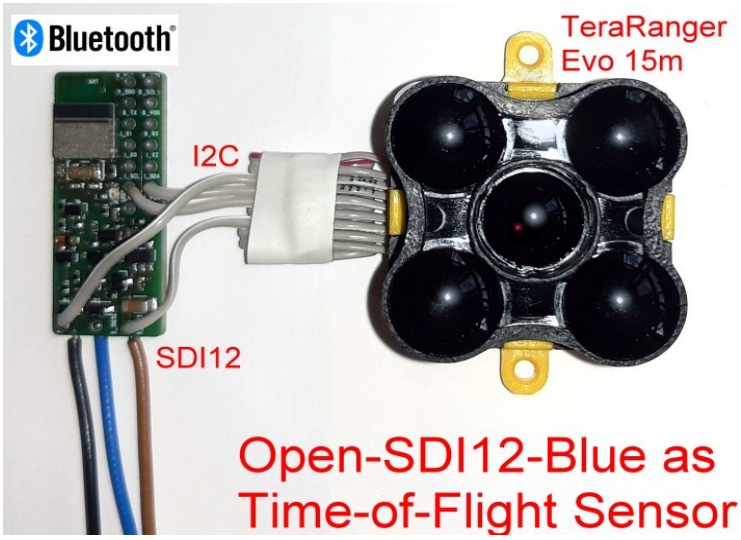
The advantages of this CPU are impressively diverse:

- Free, first-class SES development environment for the nRF52 SoCs from SEGGER6
- Very popular SoC with excellent libraries
- Very high computing power, thanks to ARM Cortex-M4F-Core @ 64 MHz
- the SoC nRF52 contains Bluetooth 5
- Driver for web bluetooth (Javascript / HTML) available



Picture 5: Core of the Open-SDI12-Blue project

One of the example implementations is Open-SDI12-Blue as a distance sensor. A high-quality sensor from the French company TeraBee is used, which is ideally suited for measuring distances to (almost) all types of solid materials. The sensor is connected via the I2C bus:



Picture 6: An SDI-12 distance sensor

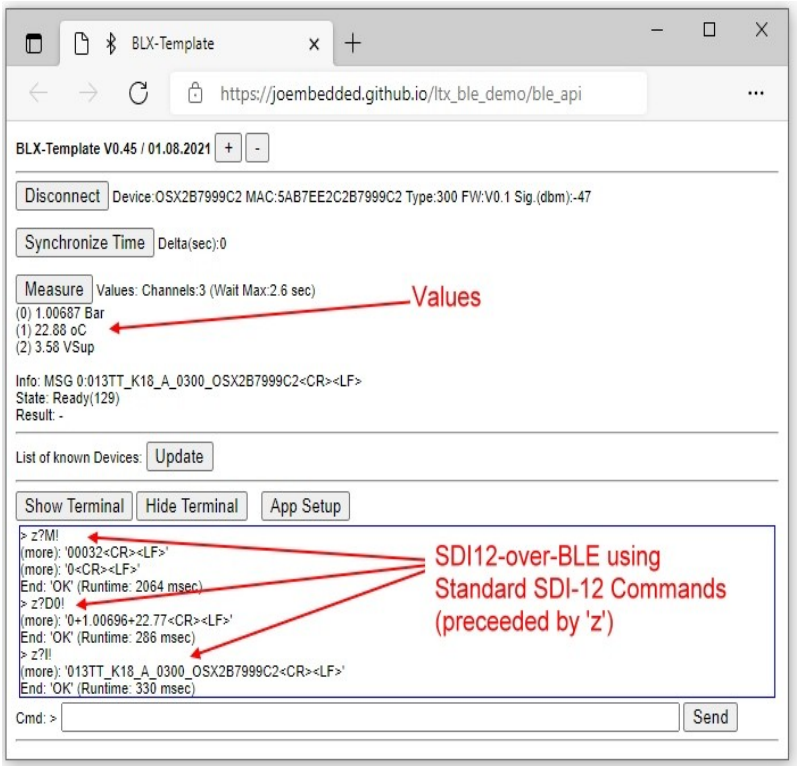
Way 1: The cable version

The specifications of SDI-12 also include a sketch for a signal driver. However on a 5V basis. The author presents a tried and tested system for a driver that also works with a 3V supply without any problems, but is also (quite well) compatible with the 5V signal standard. The only restriction is that the 3V version makes the distinction H / L or 1/0 at approx. 1.5V and not, as in the original specs, at approx. 3.5V.

In sleep mode, Open-SDI12-Blue needs <10 µA and the supply voltage may be 2.8V - 14V.

Way 2: "SDI-12 over Bluetooth"

An even more elegant (or simply an additional) way is to send SDI-12 commands via Bluetooth! An APP is not even necessary for this, because Bluetooth can now also be controlled via "simple" Javascript / HTML websites. Even firmware can be uploaded via Bluetooth if necessary. Sending SDI-12 commands via Bluetooth is done in exactly the same format as via cable!



Picture 7: "SDI-12 over Bluetooth" via Javascript / HTML

For example, local settings can be easily made via Bluetooth while a data logger communicates with the same sensor at the same time via cable! Corresponding examples and an API are contained in the documentation for "Open-SDI12-Blue".
An active Bluetooth connection only needs approx. 20-30 µA!

***

Links:

[1] SDI-12 Support Group: https://www.sdi-12.org

[2] The Open-SDI12-Blue-Project: https://github.com/joembedded/Open-SDI12-Blue
[3] A simple SDI-12 Terminal for PC: https://github.com/joembedded/SDI12Term

[4] A small list of commercial IoT Dataloggers with SDI-12:
https://www.terratransfer.org
http://geoprecision.com
https://www.ecotech.de

[5] Bluetooth SIG: https://www.bluetooth.com

[6] The free nRF52 SES - Segger Embedded Studio: https://www.segger.com

[7] Contact: joembedded@gmail.com

***