Application note

Recommendations for a robust I2C communication

Melexis
INNOVATION WITH HEART

Contents

Application note

Recommendations for a robust I2C communication

1. Scope

The Inter-Integrated Circuit (I2C) bus is a popular communication protocol used for short-distance connections between electronic devices. It's a two-wire, half-duplex, bidirectional system that uses a serial data line (SDA) and a serial clock line (SCL). Each device on the bus has a unique address, allowing a single master device to communicate with multiple slaves. Its efficiency and minimal wiring make it a common choice in consumer electronics and industrial equipment.

Key Advantages and Limitations

While the I2C bus is highly useful, it has some limitations:

- **Speed**: It's slower than other protocols like SPI, which can be a drawback for high-speed data transfer.
- **Data Integrity**: It lacks built-in checksums or error-checking algorithms like CRC or ECC, which can compromise data integrity.
- **Bus Length**: The length is limited by capacitance, which can affect signal integrity over longer distances.
- **Address Conflicts:** The limited number of unique addresses (127 or 1023) can be an issue in complex systems with many devices.
- **Additional Components:** It requires pull-up resistors for its open-drain outputs.
- **Noise Susceptibility:** It can be vulnerable to electromagnetic interference from components like inverters and DC-DC converters.

These challenges require careful design to ensure the bus is reliable and performs well in a given application. This document aims to provide methods for achieving reliable I2C communication, even in harsh, noisy environments.

For in depth information on I2C bus – check the specification [1]: https://www.nxp.com/docs/en/user-guide/UM10204.pdf

2. Hardware measures

There are many players that may render the operation of an I2C bus intermittent. Probably everyone system designer has his own “secret sauce” to make it work.

Below, we give a list with the major contributors, which, by no means, pretend to be completely exhaustive.

2.1. Pull-up resistors – value and placement

From the physical layer, the I2C bus is designed to be driven only through open-drain (open collector) connections pulling the bus down, it is pulled up by a pair of pull-up resistors - one on the clock (SCL) line and one on the data (SDA) line. The EMC engineer will say: “I want strong pull-ups!”

The power management will say: “No, no. Weak pull-ups for me!”.

And both are right.

/Note: “strong” is typically associated with a resistor with a low-enough value; “weak” – is a resistor with high-enough value/.

We have seen engineers struggle with unreliable I2C communication due to either the entire lack of or incorrect pull-up resistors. Yes, it is possible to actually communicate even without the resistors due to parasitic pull-ups which will be much larger than required, meaning that it will pull weakly enough to get the bus High and under some conditions can provoke an ACK (acknowledge) from a slave device.

When we check with [1], we can find lowest and highest values:

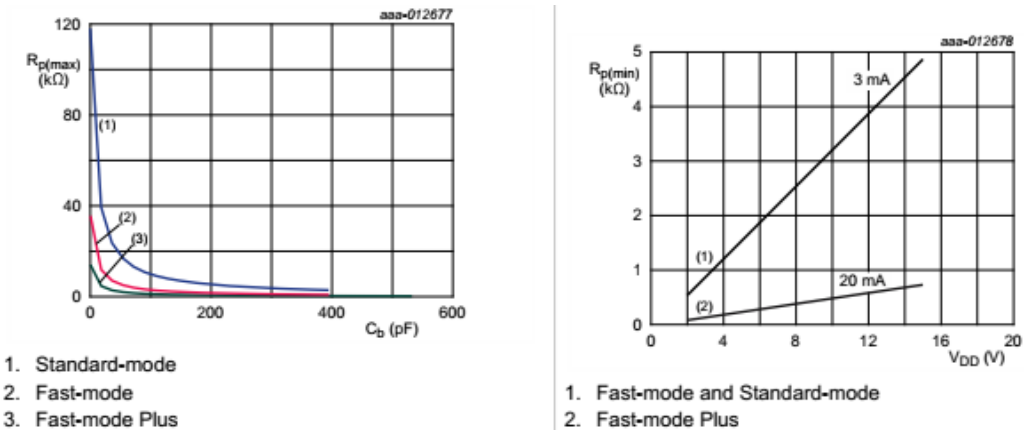

Figure 1 Limits for Rpu acc. [1]. Note: V_{DD} in this graph is the pull-up voltage Vpu

There is no clear specification of what the size of these pull-up resistors should be, and for good reason, but this causes a lot of uncertainty. The I2C specification does specify that the maximum bus capacitance should be 400pF. This is a pretty tricky requirement to meet if you have a large PCB with a number of devices on the bus and it is often overlooked, so it is typical to encounter boards where the capacitance is exceeding the official specification. In the end, the pull-up needs to be strong enough (that is small enough) to pull the bus to V_{PU} fast enough to communicate at the required bus speed (typically 100kHz to 1MHz).

When we elaborate further, we get the SCL/SDA I/O characteristic:

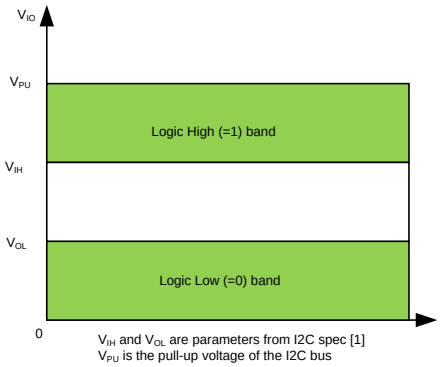

Figure 2 I/O Characteristic

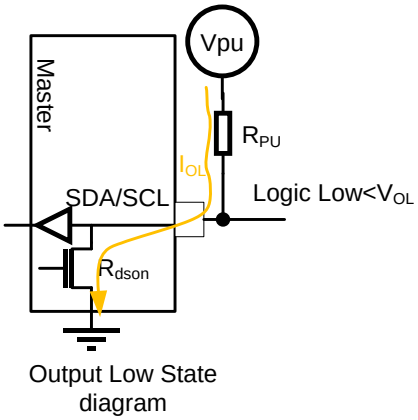

Figure 3 Output Low

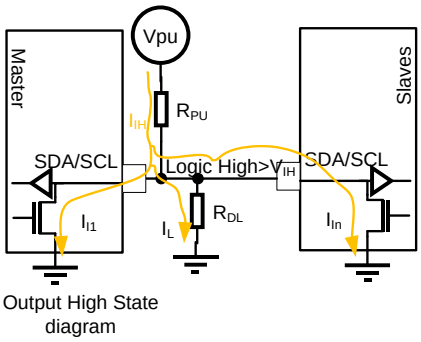

Figure 4 Output High

So when the output voltage of either pins is between 0V and V_{OL} , the corresponding connected input will interpret is as Logic “0”. The target is to be as close as possible to 0V(=Ground), in order to have as much as possible margin for noise immunity. The exact voltage level is defined by the drain-source resistance of the output driver (Rdson), as well as the R_{PU} (see Fig.3).

When the output voltage of either SCL or SDA is between V_{PU} and V_{IH}, then the signal will be interpreted as Logic “1”. The target is to be as close as possible to V_{PU}, in order to have as much as possible margin for noise immunity. The exact level is defined by the total High-Level Input Current (I_{IH}), see Fig.4. Even when no device is pulling down the line and it is a logical high, current continues to flow through the pull-up resistors. This current is caused by the leakage of the digital inputs of the devices on the bus I_{I1} .. I_{In}, leakage from low quality PCB materials and possibly from soldering residues, indicated as R_{DL}. Some of these cannot be foreseen, but, assuming quality materials and good manufacturing practices, the input pin leakage is dominant.

In the band between V_{OL} and V_{IH}, the signal may be interpreted as either 1 or 0.

With a weak Rpu, there will be a strong “0”, while a strong Rpu we get a strong Logic High level.

Keeping the lines short and minimal coupled to each other and their surroundings would be the solution. If you are running other data lines over/under the I2C lines, arrange for them to cross seldom and at 90-degree angles. Long bus wires with a weak resistive pullup are almost as good as an antenna.

So all above is to say that the sweet spot value is not a constant and varies with bus capacitance, clock speed, driver capability.

When the pull-up resistors are selected, a question comes where they should be placed. Figure 5 below shows placement at the master side, Figure 6 – at the slave side.

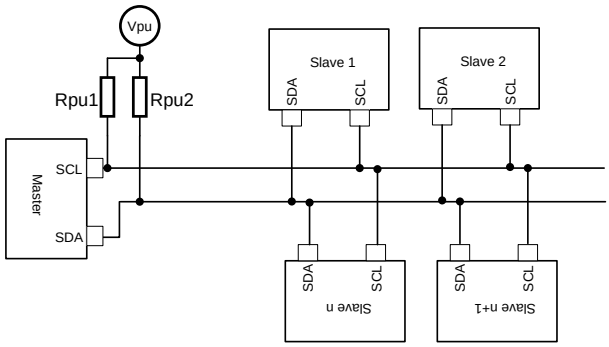

Figure 5 Pull-ups at master

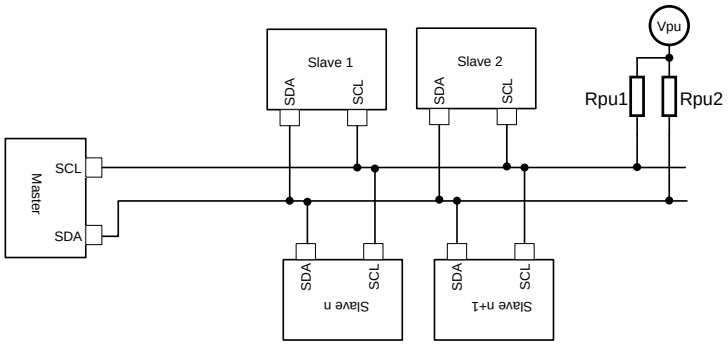

Figure 6 Pull-up at slave side

Bus speed vs. power consumption. The pull-up resistors must be reduced in value, when increasing the bus speed or when there is significant bus capacitance. The lower value resistors cause increased current draw, as each logical low on the bus creates a path to ground, negatively impacting power consumption. The bus speed can become a trade-off between completing tasks quickly and returning a system to a low-power idle state, versus the additional current draw created by the higher bus speed requirements.

2.2. Bus capacitance, additional capacitors – value and placement

“Capacitance, capacitance, capacitance.” – advises the EMC expert;

“Maximum 400pF” reminds the I2C specification.

In noisy environment, what inverters are, effectively using this maximum allowed bus capacitance is often the easiest way to get a reliable I2C communication. And even when there are specific schematic techniques to artificially increase these 400pF up to several nanofarads by using specific bus accelerators and/or splitting the bus into branches, adding a small discrete component at the sensitive nodes may help.

When any of the SDA/SCL signals is in Logic High state, the bus impedance is mainly defined by the pull-up, thus pretty high. If the BOM (Bill Of Materials) allows, it might be beneficial, to add a dedicated capacitor at everyone node on the bus, as shown on Figure 7 below. This alters the bus impedance and helps avoid signal reflections when a HF noise is coupled.

If we look into the oscilloscope plot of a typical I2C communication – Figure 8 below – several phenomena can be noticed. These can turn into issues in certain conditions:

a) The falling edges (e.g. High-to-Low transition) are almost instantaneous. The reason is that those are defined by the built-in transistors in the Master/Slaves, which have very low resistance R_{dson} , typically a few tens of Ohms.

b) The rising edges (e.g. Low-to-High transition) take some time. The root cause is the time constant defined by the pull-up resistors Rpu and the sum of all parasitic and discrete capacitors C_{SDA} (or C_{SCL}) on either of the traces.

c) The glitches that appear on both SDA/SCL signals (and do not exist in software).

The root cause of these glitches is often the trace-to-trace parasitic coupling capacitance – $C_{SDA-SCL}$, indicated with red color on Figure 7. The closer are SDA and SCL, the higher is the crosstalk between the traces. The faster dV/dt of the signals is, the higher are the glitches.

These glitches may generate false START or STOP conditions in neighbor devices on the line.

The cure is either apply the 3W rule, or add a guard trace between SDA and SCL. The 3W rule states that a spacing of at least 3 times the width of the two traces effectively reduces the coupling between them.

Adding a guard Ground trace between SDA and SCL also helps. Note that the parasitic C_{SDA} and C_{SCL} will also increase.

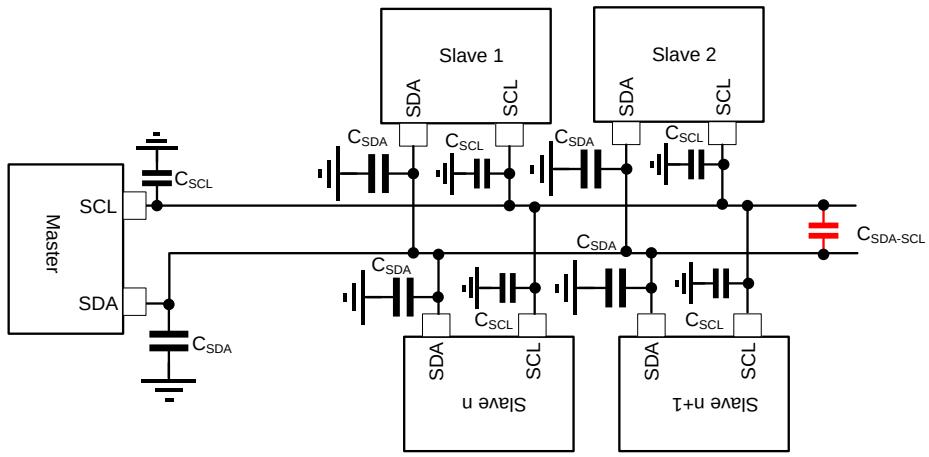

Figure 7 Capacitor placement

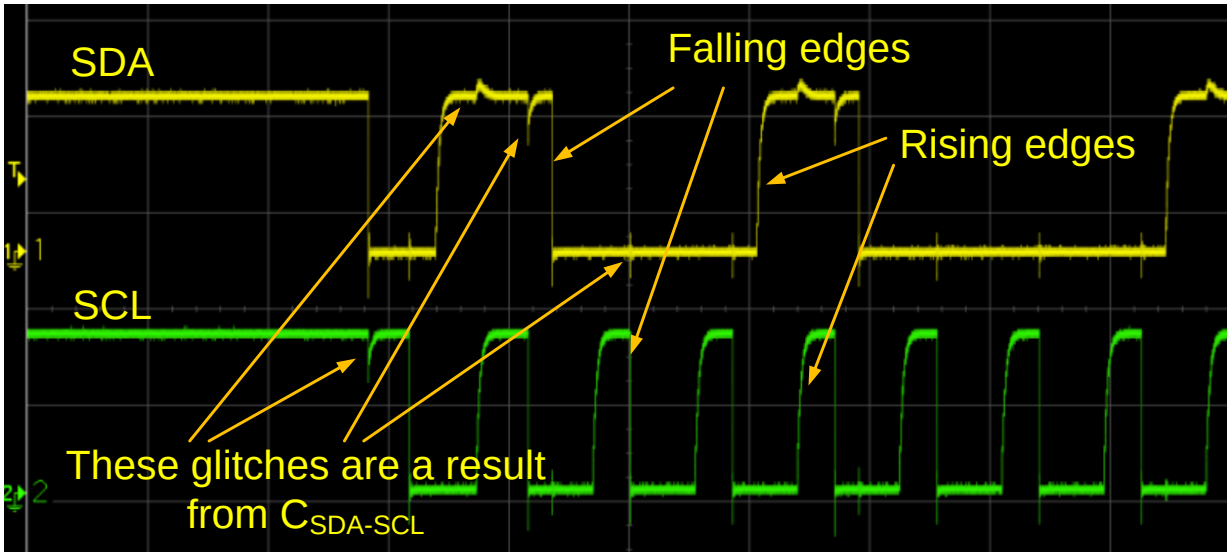

Figure 8 A typical I2C communication

When the PCB stack allows, it is beneficial to properly shield the I2C lines from the disturbing noise sources.

This is one technique we use and show results in Section 4.

2.3. Bus isolators

In the classic I2C bus all SDA-s are connected together and all SCL pins too. I2C buses aren't always the most robust, being vulnerable to single point failures where one bad part takes down other parts of the bus. Robust design is a key, so that even in the event of a failure in one component, the rest of the hardware can keep working (Figure 9). This fault can be purely hardware-related, as faulty component- a shorted capacitor, resistor or IC, or solder residue, or dirt, or tin whiskers.

Tin whiskers are a phenomenon that occurs spontaneously from the plated surface in electrical devices when metals form long whisker-like projections over time. "Tin Whiskers" are typically no more than a few millimeters long, but even that is enough to cause considerable problems.

A software flaw can lead to stuck in SDA or SCL signals, thus improper timings.

When reliability and robustness are a must, the bus isolators come into the picture. As bus isolator can be used simple discrete schematic solution, or an integrated one. The operating principle is that when either SDA or SCL are stuck low longer than a certain time, the corresponding device is automatically disconnected. Shown on Figure 10, the bus isolation can be applied on a device level and also a bus branch level, depending on the demand.

Some more sophisticated bus isolators provide a feedback signal to the Master when a fault occurs and even may try to recover a stuck bus by applying automatic clock pulses (this is one technique to force a Software reset of a Slave, described in the next Chapter).

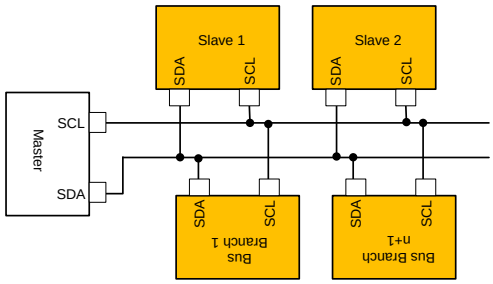

Figure 9 One fails – all fail

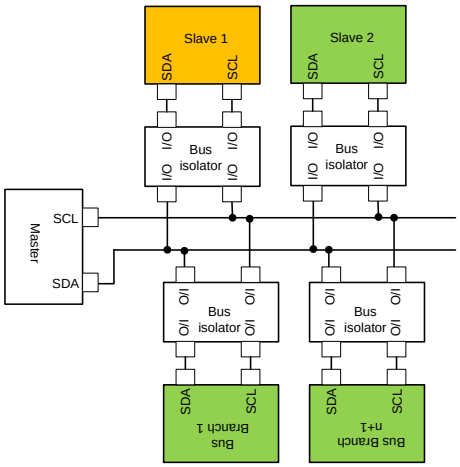

Figure 10 Using I2C bus isolators

2.4. Hardware reset

Software reset is reviewed in the next chapter

When the Master or a Bus isolator reports a bus fault condition, the Master may try to recover the operation by resetting the corresponding Slave(s).

Hardware reset may be implemented by a dedicated Reset signal, or toggling the supply voltage on a specific slave device or a specific branch of the I2C bus, as shown on Figure 11 below.

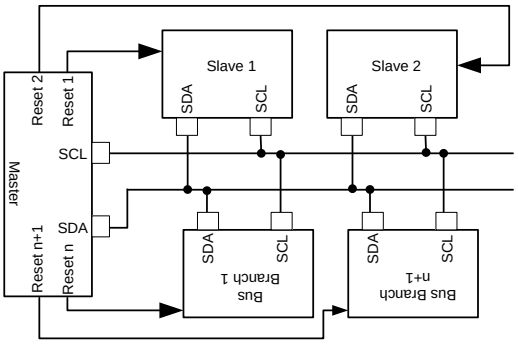

Figure 11 Implementation of hardware reset

From hardware perspective, a simple Master-controlled analog switch can be used.

2.5. Supply filtering

The I2C is often advertised as a two-wire bus, thus the importance of the supply system is under looked.

For a remote sensor, the power integrity is as important as the signal integrity. A transient or AC disturbance coupled at the supply pins can be as dangerous as a false STOP condition on the bus, because it can lead to malfunction, undesired RESET or stuck in a false state.

In our Melexis datasheets, we often give a typical application schematic that is very minimalistic. Sometimes there is another one, recommended for noisy and harsh environments. The simplest to cope with noise is to add more (a higher) supply by-pass capacitor and/or a dedicated L-C filtering, when the application allows, as shown on the figure below.

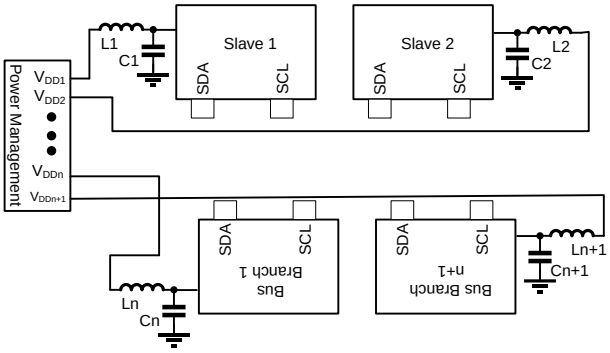

Figure 12 Supply filtering

3. Measures in software

There is a number of measures that should be implemented in software different faults , for example bitflips, delay, change, and lock, to get a reliable I2C.

3.1. Software reset

Even though a dedicated hardware reset is preferred, when available, the I2C bus specification in Section 3.1.16 defines a method to clear bus - if the data line (SDA) is stuck LOW, the master should send nine clock pulses. The device that held the bus LOW should release it sometime within those nine clocks. If not, then use the HW reset or cycle power to clear the bus. This procedure can be organized in such way that the Master examines the SDA bus and exits when the bus is released.

Obviously, the above procedure works only for the data line. In the unlikely event where the clock (SCL) is stuck LOW, the preferential procedure is to reset the bus using the HW reset signal if your I2C devices have HW reset inputs. If the I2C devices do not have HW reset inputs, cycle power to the devices to activate the mandatory internal Power-On Reset (POR) circuit.

An unscheduled master restart may also cause an SDA lock in low state. When an unscheduled master reset happens, some slaves become unsynchronized, then transmit faulty values, in which it can be overcome by having a scheduled reset to the slave devices after the master's reset.

3.2. Watchdog (timeout)

Using a watchdog timer or timeslots during I2C communication is another key feature where increased reliability is required. This is a simple, yet effective means to avoid program blocks, waiting for the response of the unresponsive sensor, etc. The Watchdog Timer (WD) operates by continuously monitoring the microcontroller's activity. If the system fails to reset the WD within a predefined time, it assumes there's a fault (like an infinite loop or unresponsive state) and automatically resets the system to a safe state.

In the I2C communication, there are certain parameters that can help configuring the WD timeout interval. These are Byte width—all bytes are 8 bits wide, with no exceptions, message length—technically there is no

maximum length for a message; a minimum message consists of 2 bytes (an address and a data byte), however for a certain data requests, the response can be time-mapped.

3.3. Bus speed (SCL frequency)

A properly designed I2C bus operates at the speed of the slowest slave.

The more slaves there are on the bus, the more likely you will be to have an increased bus capacitance and signal integrity problems.

Using lower than the highest possible SCL frequency allows for a longer settling time of the signals and also lower EMI emissions from the application.

3.4. Handling NACKs and/or incomplete messages

Every byte sent on the I2C line is acknowledged (positively or negatively) by the slave or master that just received the last byte. Its arbitrage purpose is for the device to indicate “OK, proceed further”.

When the master is reading bytes from a slave, the master acknowledges each one before the next byte is sent. If the master is writing bytes to the slave, it’s the slave that has to acknowledge. A low voltage level, pulling the data line down, is the positive ACK signal.

Depending on the context, there could be negative-acknowledge (NACK) signal.

Missing Acknowledgment (ACK) is one of the main causes of bus lockups, they can occur due to various reasons: timing delay between SCL and SDA, missing/unexpected SCL pulses, incomplete 8-bit block, and missing bytes.

4. Test results

We have tested the EMC susceptibility on a 4-layer PCB. The SCL and SDA are routed on inner layer, while the other layers are configured as ground shield. Additionally, we have implemented ferrite beads on SDA and SCL. With such configuration we pass the tests.

5. Disclaimer

The content of this document is believed to be correct and accurate. However, the content of this document is furnished "as is" for informational use only and no representation, nor warranty is provided by Melexis about its accuracy, nor about the results of its implementation. Melexis assumes no responsibility or liability for any errors or inaccuracies that may appear in this document. Customer will follow the practices contained in this document under its sole responsibility. This documentation is in fact provided without warranty, term, or condition of any kind, either implied or expressed, including but not limited to warranties of merchantability, satisfactory quality, non-infringement, and fitness for purpose. Melexis, its employees and agents and its affiliates' and their employees and agents will not be responsible for any loss, however arising, from the use of, or reliance on this document. Notwithstanding the foregoing, contractual obligations expressly undertaken in writing by Melexis prevail over this disclaimer.

This document is subject to change without notice, and should not be construed as a commitment by Melexis. Therefore, before placing orders or prior to designing the product into a system, users or any third party should obtain the latest version of the relevant information.
Users or any third party must determine the suitability of the product described in this document for its application, including the level of reliability required and determine whether it is fit for a particular purpose.

This document as well as the product here described may be subject to export control regulations. Be aware that export might require a prior authorization from competent authorities. The product is not designed, authorized or warranted to be suitable in applications requiring extended temperature range and/or unusual environmental requirements. High reliability applications, such as medical life-support or life-sustaining equipment or avionics application are specifically excluded by Melexis. The product may not be used for the following applications subject to export control regulations: the development, production, processing, operation, maintenance, storage, recognition or proliferation of:
1. chemical, biological or nuclear weapons, or for the development, production, maintenance or storage of missiles for such weapons;
2. civil firearms, including spare parts or ammunition for such arms;
3. defense related products, or other material for military use or for law enforcement;
4. any applications that, alone or in combination with other goods, substances or organisms could cause serious harm to persons or goods and that can be used as a means of violence in an armed conflict or any similar violent situation.

No license nor any other right or interest is granted to any of Melexis' or third party's intellectual property rights.

If this document is marked "restricted" or with similar words, or if in any case the content of this document is to be reasonably understood as being confidential, the recipient of this document shall not communicate, nor disclose to any third party, any part of the document without Melexis' express written consent. The recipient shall take all necessary measures to apply and preserve the confidential character of the document. In particular, the recipient shall (i) hold document in confidence with at least the same degree of care by which it maintains the confidentiality of its own proprietary and confidential information, but no less than reasonable care; (ii) restrict the disclosure of the document solely to its employees for the purpose for which this document was received, on a strictly need to know basis and providing that such persons to whom the document is disclosed are bound by confidentiality terms substantially similar to those in this disclaimer; (iii) use the document only in connection with the purpose for which this document was received, and reproduce document only to the extent necessary for such purposes; (iv) not use the document for commercial purposes or to the detriment of Melexis or its customers. The confidentiality obligations set forth in this disclaimer will have indefinite duration and in any case they will be effective for no less than 10 years from the receipt of this document.

This disclaimer will be governed by and construed in accordance with Belgian law and any disputes relating to this disclaimer will be subject to the exclusive jurisdiction of the courts of Brussels, Belgium.

The invalidity or ineffectiveness of any of the provisions of this disclaimer does not affect the validity or effectiveness of the other provisions.
The previous versions of this document are repealed.

IATF 16949 and ISO 14001 Certified