AN5988

Application note

TSC1641: I3C capabilities

Introduction

The TSC1641 is a current, voltage, DC power, and temperature monitoring analog front-end (AFE). It has a double ADC-path for current and voltage to ensure the most accurate DC power computation. Programmable conversion timing and configuration modes make the device flexible to use and suitable for many systems. Alert signals can be raised on a dedicated pin upon several conditions.

The TSC1641 is a MIPI I3C™ SDR target that complies with all the requirements of the BASICS V1.1.

The MIPI I3C™ is a modern bus, which has useful features such as in-band-interruption and dynamic address assignation.

This application note gives the basic information to start using the TSC1641 on a MIPI I3C™ bus.

1 Glossary and acronyms

I3C: MIPI improved inter-integrated circuit interface.

I²C: Inter-integrated circuit. Two-wire communication protocol for connecting multiple devices, allowing data exchange and control within electronic systems.

SCL: Clock line.

SDA: Data line.

IBI: In-band interrupt.

An in-band-interrupt is an interruption triggered by a target. The target initiates a start request while the I3C bus is idle, by pulling the SDA low while the SCL is high.

The controller can then either proceed to a Nack or launch a clock pattern to give the target the possibility to talk.

HJ: Hot-join.

A hot-join is an optional feature that a target can have. A hot-join request is carried out by a target, which is not integrated in the I3C bus, to ask to the controller to join the bus.

CCC: Common command code.

A set of CCCs are available and described in the basic specifications of the I3C bus written by the MIPI alliance.

Each CCC can ask or write to a device for specific information.

Not all CCCs are mandatory for a target or a device.

2 Introduction to I3C™ and comparison with I²C

MIPI I3C™ is a standardized bus, managed by the MIPI alliance. Its specifications are aimed at becoming a standard in most use-cases such as the automotive industry, internet-of-things, and manufacturing.

The goal of this bus is to improve and ease sensor integration in systems. It allows the development of more complex and intelligent devices to be implemented in systems by having flexible features.

2.1 Comparison with I²C

I3C (MIPI I3C) offers several advantages over its predecessor I²C (inter-integrated circuit). The following are some reasons why I3C is considered better than I²C:

Enhanced data transfer rates: I3C provides significantly higher data transfer rates compared to I²C. While I²C typically operates at speeds up to a few hundred kilobits per second, I3C supports faster data rates up to 12.5 MHz.

Backward compatibility: I3C maintains backward compatibility with I²C, meaning I²C devices can be easily connected to an I3C bus without requiring any modifications. This compatibility ensures a smooth transition from I²C to I3C, allowing existing I²C devices to coexist with newer I3C devices on the same bus. This feature makes I3C a viable and convenient choice for system upgrades.

Reduced pin count and power consumption: I3C reduces pin count and power consumption compared to I²C. I3C does not require pull-up resistors. Moreover, I3C supports an in-band-interrupt feature so that an additional “alert” pin is not used. See Section 2.2.3.

Dynamic addressing and device enumeration: With I²C, device addresses are assigned statically, limiting the number of devices that can be connected to the bus. In contrast, I3C supports dynamic address assignment, allowing devices to be added or removed from the bus without conflicts. This flexibility enables scalability and simplifies system integration.

Improved bus management and control: I3C introduces advanced bus management and control features, such as hot join detection, and multi-controller support. These features enhance bus reliability, enable efficient power management, and facilitate seamless communication between devices. I²C lacks these capabilities, making I3C a more robust and versatile interface for complex system architectures.

Standardized protocol: I3C is an industry-standard protocol developed by the MIPI alliance, ensuring interoperability and widespread adoption among different manufacturers. This standardization promotes compatibility, simplifies integration, and enables a broader ecosystem of I3C-compliant devices and components.

In summary, I3C offers higher data transfer rates, backward compatibility with I²C, reduced pin count and power consumption, dynamic addressing, advanced bus management features, and standardized protocol. These advantages make I3C a more capable and future-proof interface, suitable for demanding applications requiring faster and more efficient communication between devices.

Table 1. Comparison between I3C and I²C

Feature	I3C	I²C
Number of wires	2 (data & clock)	2 (data & clock) + 1 for interrupts
Communication speed	12.5 MHz in SDR mode	Typically 400 kHz - 1 MHz
Bidirectional	Yes	Yes
Pull-up resistors needed	No	Yes
Dynamic addresses	Yes	No
Broadcast communication	Yes	No
In-band interrupts	Yes	No

2.2 Main MIPI I3C™ features implemented in the TSC1641

MIPI I3C™ embeds several features requested for modern applications, and many of these are implemented in the TSC1641:

- Support of 3.3 V
- Backward compatibility with I²C
- Dynamic address only (the controller does not need to know the hardware address of the TSC1641)
- Passive hot-join (the TSC1641 asks to join the I3C bus if it sees an I3C communication)
- In-band-interrupts (the TSC1641 can trigger alerts directly on the I3C bus)

2.2.1 Communication speed

I3C frames are made of an open-drain part and a push-pull part.

The TSC1641 is able to communicate in I3C SDR mode only and up to 12.5 MHz.

Table 2. Communication speeds

Communication type	Speed
Open drain	1 MHz
Push-pull	12.5 MHz

2.2.2 Accepted CCC commands

Moreover, the TSC1641 can understand the majority of CCC commands.

In Table 3. List of available CCCs the most important CCC commands accepted by the TSC1641 are described.

Table 3. List of available CCCs

Command name	Command code Broadcast / direct	Default	Description
ENEC	0x00 / 0x80		Enables target IBI capability.
DISEC	0x01 / 0x81		Disables target IBI capability.
RSTDAA	0x06 / N.A.		Forgets current dynamic address and waits for a new assignment or an I²C communication
ENTDAA	0x07 / N.A.		Enters controller initiation of target dynamic address assignment. If the target already has a dynamic address assigned there is no impact.
SETDASA	N.A. / 0x87		The controller assigns a dynamic address to a target with a known static address. In the case of the TSC1641, the static address is 10000xx depending on the A0/A1.
SETAASA	0x29 / N.A.		The controller tells every target with a static address to use its static address as its dynamic address.
SETNEWDA	N.A. / 0x88		The controller assigns a new dynamic address to any I3C target (only if ENTDAA is supported).
GETPID	N.A / 0x8D	0x0208020A0001	Gets the target's provisional ID (ENTDAA supported). Bit 3 varies according to A0/A1 pins (see Table 7).
GETBCR	N.A / 0x8E	0x02	Gets a device's bus characteristics register.
GETDCR	N.A / 0x8F	0x00	Gets a device's device characteristics register.
GETSTATUS	N.A / 0x90		Gets a device's operating status.
RSTACT	0x2A / 09A		Configures and queries target device reset action and timing. 0x00: no action on the target after the reset pattern. 0x01: I3C reset only. 0x02: equivalent to software reset.
SETGRPA	N.A. / 0x9B		The host assigns a group address to a device.
RSTGRPA	0x2C / 0x9C		The host removes the target device from an indicated group address by resetting the assigned group address.
GETCAPS	N.A. / 0x95		The host asks the target device what optional capabilities it supports. (1)

1. *NA: Some CCCs do not exist in direct or broadcast format.*

2.2.3 In-band interruptions (IBI)

The TSC1641 has several alerts that can be activated.

These alerts are usable in I²C and I3C. In I²C the alerts are monitored by the designated alert pin.

In I3C, the use of the alert pin may be avoided due to the IBI feature. In I3C, when an alert is generated by the component, it rises directly on the bus.

The TSC1641 can trigger alerts directly on the I3C bus.

These alerts are set by the mask register as seen in Table 4 and limits registers (registers 0x09 to 0x0E). It is possible to enable or disable different kinds of alerts by writing into the register.

Table 4. **Mask register (0x06)**

Bit n°	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
name	SOL	SUL	LOL	LUL	POL	TOL	CVNR								APOL	ALEN
detail	Shunt over limit	Shunt under limit	Load over limit	Load under limit	Power over limit	Tempera ture over limit	Conversi on ready								Alert polarity	Alert latch enable
POR value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

If the device is in I3C mode (i.e. a dynamic address has been assigned to the TSC1641) and the in-band-interruption feature is activated, then an alert is triggered on the designated ALERT pin and simultaneously on the I3C bus.

In Figure 1, an IBI may be seen on the I3C bus by the TSC1641. The device gives its address (in this case the address is 08) to inform the controller. Note that the alert pin (at the bottom) and the interruption are triggered almost simultaneously.

Figure 1. **In-band interrupt initiated by the TSC1641**

2.2.4 Hot-join

Hot-join is a feature in the I3C interface that enables devices to be dynamically added or removed from the bus without disrupting the ongoing communication. It allows for seamless integration of devices in an I3C bus network, even while the bus is operational.

Hot-join brings several benefits to the I3C bus network. It allows for flexible system expansion, as devices can be added or removed dynamically during runtime without requiring system-wide resets or disruptions. It simplifies the integration process and reduces the need for manual configuration of addresses or bus parameters. Hot-join also enables the support of plug-and-play functionality, making it easier to connect and disconnect devices on the bus.

A device which is not on the bus initiates the hot-join. The device monitors the bus, observes an I3C communication on the bus, and requests to be added to the bus by pulling to low the SDA line.

2.2.5 Bus characteristics register (BCR)

The bus characteristic register is an 8-bit register describing the capabilities of the device regarding I3C.

Each bit explains a capability of the device. The table below is an extract from the MIPI I3C basic specification (public version V1.1.1). The full table is available on the member version of the I3C basic specification V1.1.1.

By sharing this register with other devices, it allows other devices to know and understand the capabilities of the device and communicate efficiently with it.

Table 5. Bit meaning for BCR value

Bit	Name	Description
BCR[7]	**Device role[1]**	2'b00: I3C target 2'b01: I3C controller capable 2'b10: reserved for future definition by MIPI alliance I3C WG 2'b11: reserved for future definition by MIPI alliance I3C WG
BCR[6]	**Device role[0]**	
BCR[5]	**Advanced capabilities**	1: Supports optional advanced capabilities Uses GETCAPS CCC to determine which ones 0: Does not support optional advanced capabilities
BCR[4]	**Virtual target support**	0: Is not a virtual target and does not expose other downstream device(s) 1: Is a virtual target, or exposes other downstream device(s)
BCR[3]	**Offline capability**	0: Device always responds to I3C bus commands 1: Device does not always respond to I3C bus commands
BCR[2]	**IBI payload**	No data bytes follow the accepted IBI 1: One data byte (MDB) follows the accepted IBI, and additional data bytes may follow; see also the set/get maximum read length CCC See also the use of IBI Payloads for timing control
BCR[1]	**IBI request capable**	0: Not capable 1: Capable
BCR[0]	**Max data speed limitation**	0: No limitation 1: Limitation

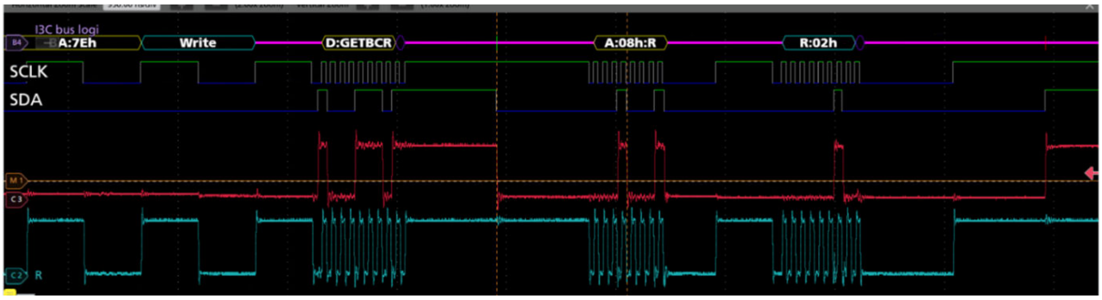
The TSC1641 BCR register contains the value 0x02 or 0b0000 0010.

The meaning for the TSC1641 is explained in Table 6.

Table 6. TSC1641 BCR value explanation

Bit	Value for the TSC1641	Meaning for the TSC1641
BCR[7]	0	I3C target only
BCR[6]	0	
BCR[5]	0	Does not support optional capabilities
BCR[4]	0	Is not a virtual target
BCR[3]	0	Always responds to I3C bus commands, even in shutdown mode
BCR[2]	0	No data bytes follow the accepted IBI
BCR[1]	1	The TSC1641 is capable to perform IBI
BCR[0]	0	There is no limitation on data speed

Figure 2. GETBCR request and response from the TSC1641 to a direct GETBCR command



2.2.6 DCR device characteristics register

The DCR (device characteristic register) is a register containing the kind of device (i.e. touch sensor, ECG, etc.).

The TSC1641 DCR value is 0x00, the ID for a generic component.

2.2.7 GETCAPS

The "GETCAPS" command in I3C (MIPI I3C) is used to retrieve the device capabilities from another I3C device on the bus. It is a command initiated by a controller device to query the capabilities of a specific target device.

When a controller device sends a GETCAPS command to a target device, the target device responds by providing information about its supported features, functionalities, and operating modes.

The GETCAPS command allows the controller device to dynamically discover and understand the capabilities of the target device. It helps the controller device to determine how to interact with the target device effectively and utilize its specific features or functionalities.

By retrieving the device capabilities through the GETCAPS command, the controller device can make informed decisions about communication settings, protocols, power management, and other parameters. This enables efficient use of the target device's capabilities and promotes optimized communication within the I3C bus network.

In summary, the GETCAPS command in I3C facilitates the exchange of device capability information between devices on the bus, enabling better coordination and use of the available features and functionalities.

The TSC1641 answers to the GETCAPS command:

- 0x00:
  - It is not HDR [1] compatible

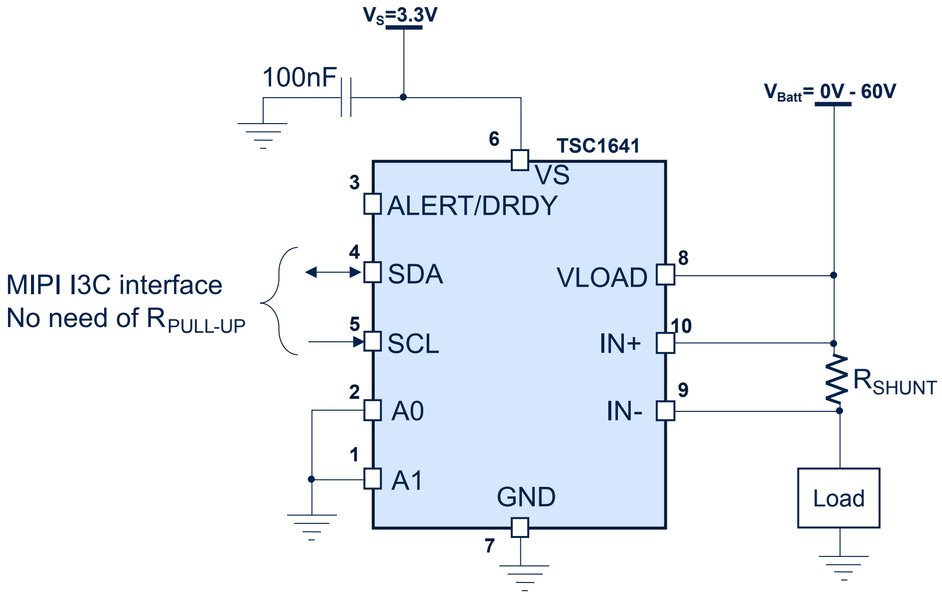
1. HDR: High data rate is an optional I3C feature.

- 0x11:
  - Can be assigned to one group address
  - TSC1641 complies with MIPI I3C Basic V1.1
- 0x18:
  - TSC1641 support optional defining byte on GETSTATUS
    - The defining byte is 0x00
  - TSC1641 support optional defining byte on GETCAPS
    - The defining byte is 0x00
- 0x00

3 Typical schematic

I3C frames are made of push-pull and open drain parts. There is no need for pull-up resistors on the boards. This reduces the number of components needed and makes the layout easier to design and reduces consumption.

Figure 3. **Typical schematic for I3C communication**



As shown, alerts are directly triggered on the I3C bus thanks to the SDA line. So, it is not necessary to connect the ALERT pin.

4 Dynamic address assignation

To enter in I3C mode, the TSC1641 must receive a dynamic address from a microcontroller.

There are several ways to assign a dynamic address to the TSC1641.

4.1 Use of static address as a dynamic address

The TSC1641 accepts the CCC SETAASA.

This command is sent to all targets connected to the bus. With this command, all the targets use their respective static address as their dynamic address.

Figure 4. **CCC SETAASA by broadcast.**

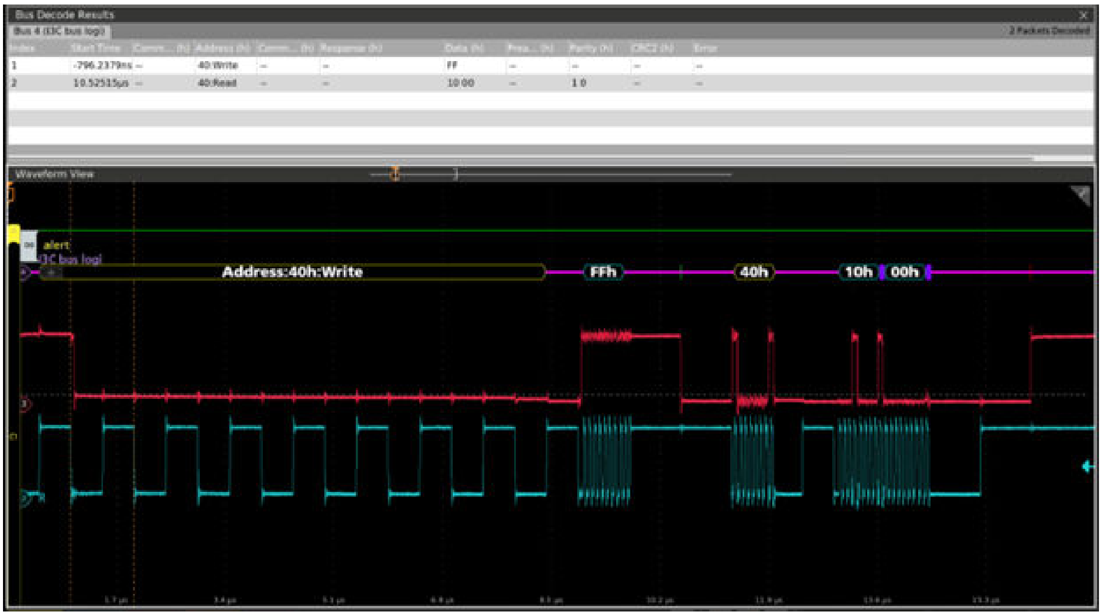
After this CCC command, the device is accessible in I3C with its static address.

The static address depends on the A0 and A1 pins.

Table 7. Target static addresses according to A0 and A1 pins

A1	A0	Target address (binary)	Target address (h)
GND	GND	1000000	40
GND	VS	1000001	41
VS	GND	1000010	42
VS	VS	1000011	43

Figure 5. In this example the static address is 0x40, after the CCC SETAASA the I3C dynamic address also becomes 0x40. This frame shows the Die ID register (0xFF register).



4.2 Assignment of a dynamic address thanks to the static address

The TSC1641 accepts the CCC SETDASA.

In other words, the TSC1641 in I²C mode can be put in I3C mode thanks to its I²C static address. The controller chooses a new dynamic address, and it is given to the TSC1641.

It differs from the SETAASA because it is a direct command (it is addressed to only one target) and the controller chooses the new dynamic address given to the target. The new dynamic address does not depend on the static address of the target.

4.3 Assignment of dynamic address thanks to bus exploration

This is an interesting and useful feature of the I3C bus.

The controller can explore the I3C bus looking for new targets to add to the bus.

For each target able to join the I3C bus, the controller proceeds into a specific dynamic address assignment process: the ENTDAA.

The controller broadcasts on the bus the CCC ENTDAA, and for each target responding to this command, a dynamic address is assigned in exchange for PID (a unique code for each target allowing to pair PID-dynamic addresses) and BCR and DCR codes (a code giving information about target capabilities). The figure below shows the ENTDAA process, in which two targets are discovered and added to the bus by the controller. During the ENTDAA process, BCR, DCR, and PID are shared by the target. In exchange, the controller assigns a dynamic address to each target (here 0x08 and 0x09).

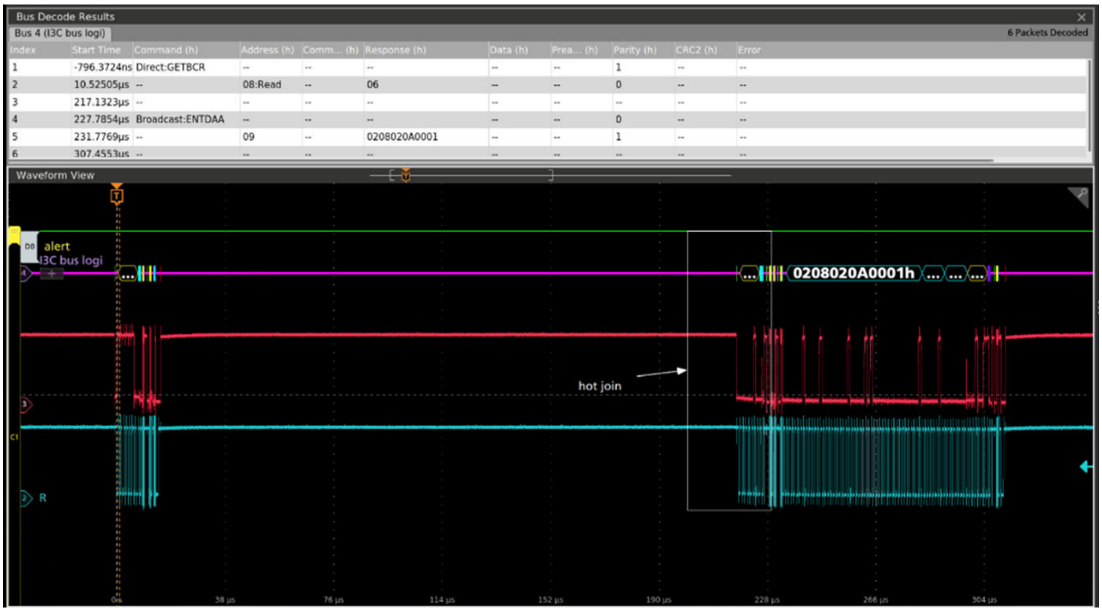
Figure 6. The ENTDAA process

4.4 Passive hot-join

Passive hot-join is a feature that allows the TSC1641 to join an existing I3C bus network without actively initiating the hot-join process. In a passive hot-join, the new device listens to the bus and detects ongoing communication before asking to join by pulling the SDA line low.

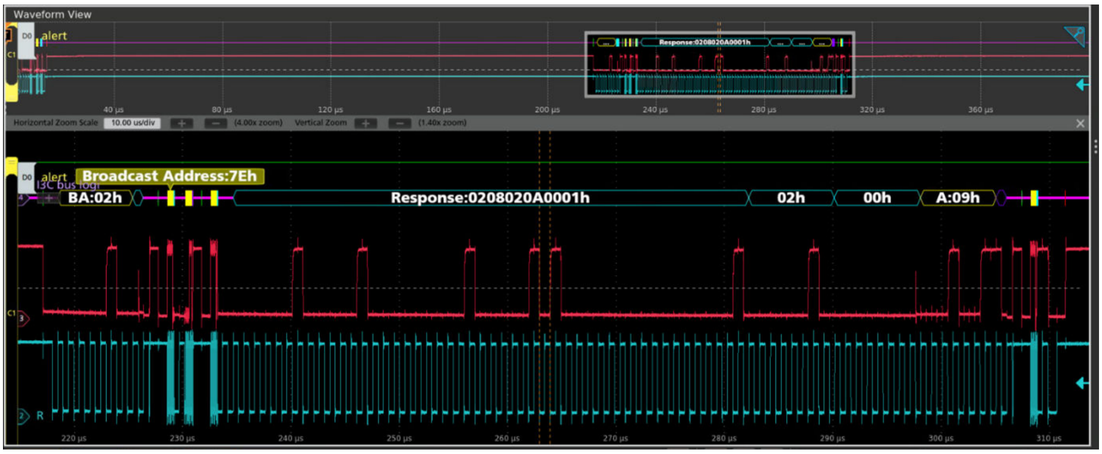
When in I²C mode, the TSC1641 continuously monitors the I3C bus for activity and waits for a trigger condition to occur. This trigger condition is a bus IDLE of 200 µs. Once the trigger is detected, the TSC1641 initiates the hot-join process, following the standard hot-join procedure as described in the I3C basic V1.1.1. In Figure 7, the first frame on the left is an I3C communication (a GETBCR, more precisely) between the controller and a device on the bus. The TSC1641 reads the I3C bus and sees this communication. After 200 µs of inactivity on the bus, the device asks to be added.

Figure 7. A GETBCR communication



In Figure 8, the TSC1641 pulls the SDA line low. The controller allows the target to communicate by launching a clock signal. Then, the broadcast address 0x02 (hot-join) is sent by the TSC1641. The controller then launches an ENTDAA process to add the device.

Figure 8. The TSC1641 pulls the SDA line to the low



5 Application to a use-case: electrical skateboard

5.1 Purpose of the demonstration

This demo was presented during the embedded world forum 2023 in Nuremberg.

Figure 9. Skateboard demo using the TSC1641 and the STM32H5 to communicate in I3C

The demo highlights the specificity in that it uses only the I3C bus to communicate.

The goal of this demo is to monitor the battery voltage of an electric skateboard battery pack and measure the current consumption of the electric skateboard.

The skateboard motor speed is controlled by a remote controller. If the speed is too high or too low, an alert must be triggered by the TSC1641 directly on the I3C bus.

The TSC1641 also uses the in-band interrupt feature to send alerts on the I3C bus when the current is too high or too low according to the thresholds set. Figure 10 shows the current and voltage measurements of the skateboard. The current is above the shunt under limit threshold of 10 mA and below the shunt over limit threshold of 200 mA, so no alert is triggered. At the bottom of the screen-shot are the current alert states. These alerts are triggered via in-band interrupts by the TSC1641 on the I3C bus.

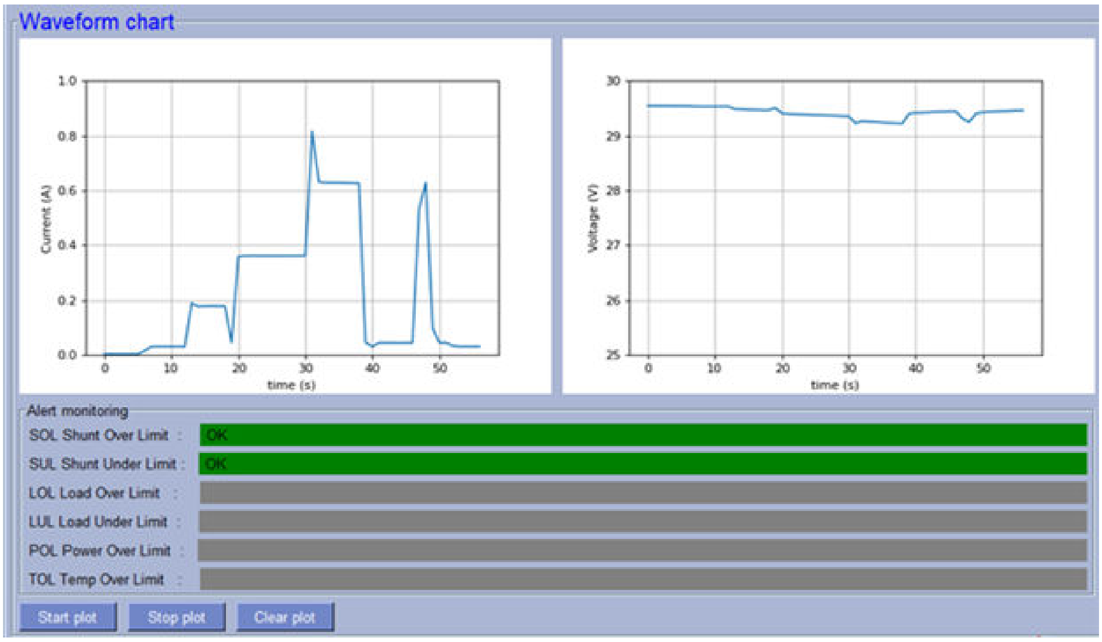
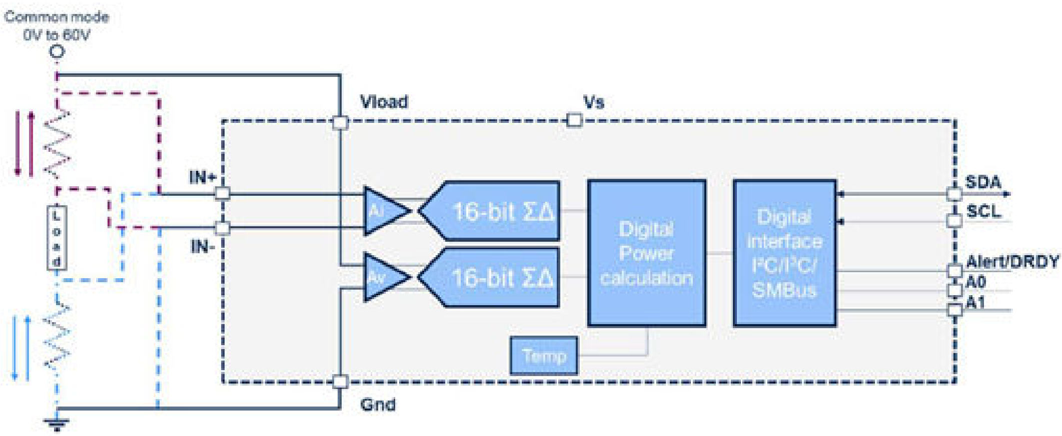


Figure 10. The current and voltage measurement on the skateboard

5.2 Block diagram

The TSC1641 (placed on the STEVAL-DIGAFEV1) has two separate modulators to monitor voltage and current.

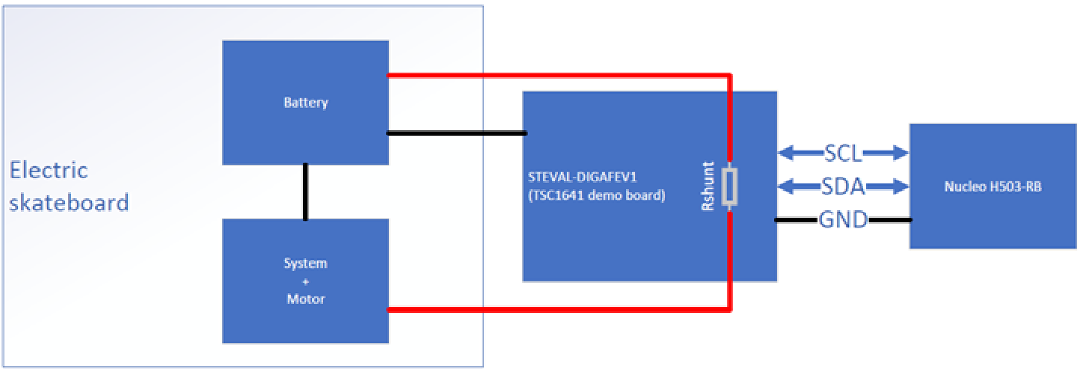
Figure 11. Schematic diagram



On the one hand, the battery is connected to the VLOAD TSC1641 input. This input is connected to the modulator to monitor voltage.

On the other hand, the current is monitored by an external shunt resistor by measuring the voltage drop surrounding the resistor by inputs in+ and in-.

Figure 12. Block diagram



5.3 Communication between the controller and the target

The bus is composed of the STM32H5 and the TSC1641.

The controller is STM35H5, which is located on the Nucleo-STM32H503RB.

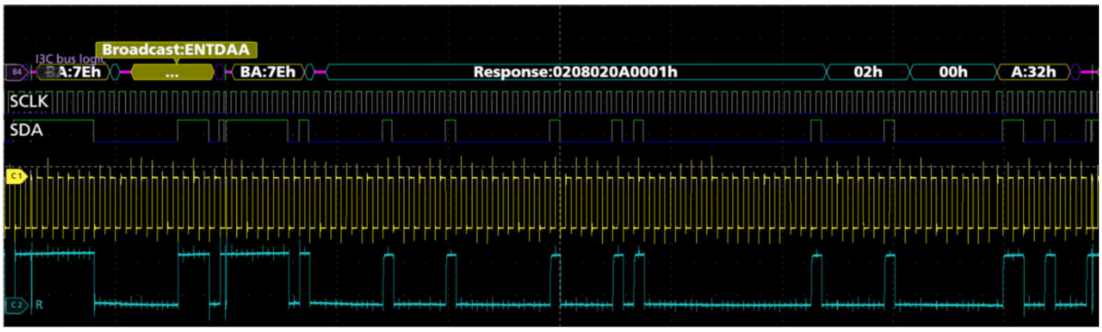
The target is TSC1641, located on the STEVAL-DIGAFEV1.

5.3.1 Addition of the TSC1641 to the I3C bus

The first step is to add the TSC1641 to the I3C bus. As previously mentioned, the TSC1641 may be added to the bus via several ways.

In the case of this demo, the method used is the ENTDAA (the controller discovers the bus and assigns an address to each target encountered).

Figure 13. ENTDAA, the STM32H5 adds the TSC1641 on the bus and gives it the dynamic address 0x32



Going forward, the TSC1641 is in I3C mode and can respond to I3C commands only.

5.3.2 Enabling the alert and setting the thresholds

At this point, the TSC1641 can communicate in I3C.

To trigger alerts on overcurrent and undercurrent, we must enable these alerts by setting the mask register.

Table 8. Mask register

Bit n°	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	SOL	SUL	OLO	LUL	POL	TOL	CNVR								APOL	ALEN
POR value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

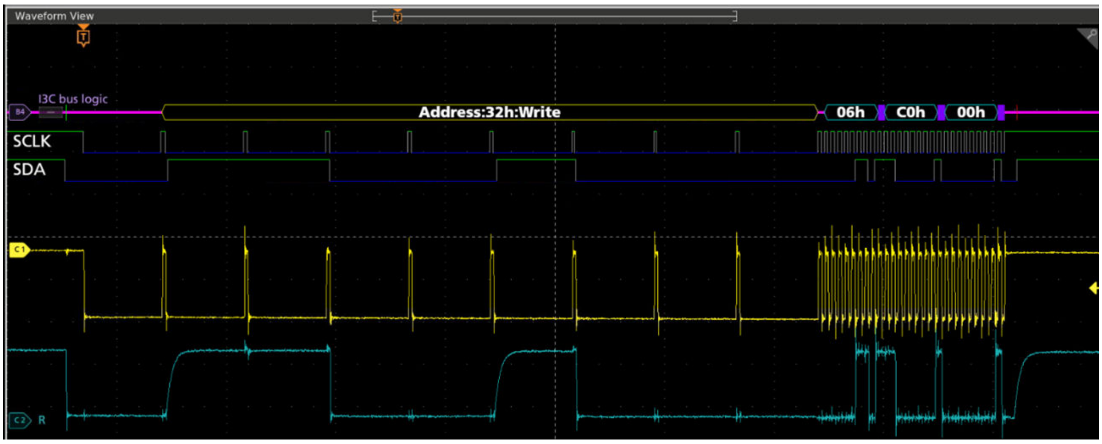
These two alerts are:

- SOL: shunt over limit

- SUL: shunt under limit

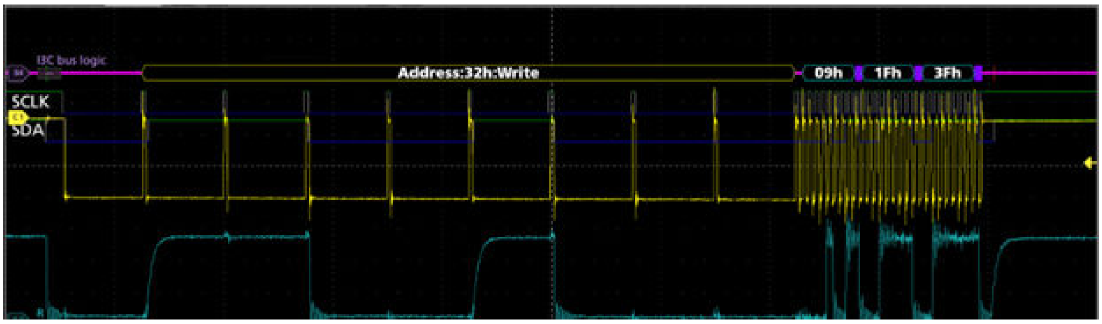
As shown in table 8, bits 15 and 14 must be set to 1 to enable SOL and SUL alerts.

Figure 14. Write on mask register



Now the alerts are activated, but the thresholds must be set.

Figure 15. Threshold written in SOL register



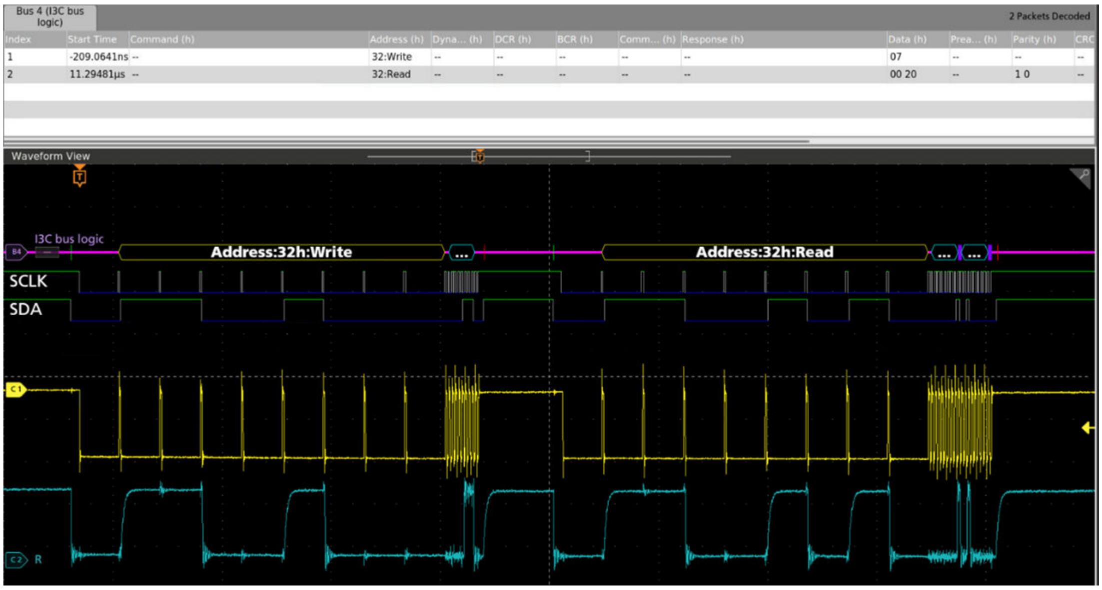
Accordingly, the TSC1641 monitors the current and triggers an in-band interrupt on the I3C when the current is too low or too high according to the previously set threshold.

5.3.3 In-band interrupt and read of flag register

When the current is too high or too low, an alert is triggered. Simultaneously, an in-band-interrupt is generated on the I3C bus (as explained in the dedicated part).

Then, the microcontroller reads the flag register to know which alert has been triggered.

Figure 16. Read of the flag register



The flag register contains the value 0x0020.

In this case, the only alert activated is SUF (shunt under limit flag), meaning that the current is under the previously set threshold.

Table 9. Flag register, each bit corresponds to an alert. If the bit = 1 an alert is triggered.

Bit n°	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name		OVF	SATF							SOF	SUF	LOF	LUF	POF	TOF	CVNF

Revision history

Table 10. **Document revision history**

Date	Version	Changes
24-Oct-2023	1	Initial release.
05-Dec-2024	2	Updated title on the cover page.

Contents

List of tables

List of figures