# Frequently Asked Questions (FAQ) for MIPI I3C® v1.0 and I3C Basic$^{SM}$ v1.0

**FAQ Version 1.0**

**8 December 2018**

MIPI Board Approved 8 December 2018

**Public Release Edition**

## NOTICE OF DISCLAIMER

The material contained herein is provided on an “AS IS” basis. To the maximum extent permitted by applicable law, this material is provided AS IS AND WITH ALL FAULTS, and the authors and developers of this material and MIPI Alliance Inc. (“MIPI”) hereby disclaim all other warranties and conditions, either express, implied or statutory, including, but not limited to, any (if any) implied warranties, duties or conditions of merchantability, of fitness for a particular purpose, of accuracy or completeness of responses, of results, of workmanlike effort, of lack of viruses, and of lack of negligence. ALSO, THERE IS NO WARRANTY OR CONDITION OF TITLE, QUIET ENJOYMENT, QUIET POSSESSION, CORRESPONDENCE TO DESCRIPTION OR NON-INFRINGEMENT WITH REGARD TO THIS MATERIAL.

IN NO EVENT WILL ANY AUTHOR OR DEVELOPER OF THIS MATERIAL OR MIPI BE LIABLE TO ANY OTHER PARTY FOR THE COST OF PROCURING SUBSTITUTE GOODS OR SERVICES, LOST PROFITS, LOSS OF USE, LOSS OF DATA, OR ANY INCIDENTAL, CONSEQUENTIAL, DIRECT, INDIRECT, OR SPECIAL DAMAGES WHETHER UNDER CONTRACT, TORT, WARRANTY, OR OTHERWISE, ARISING IN ANY WAY OUT OF THIS OR ANY OTHER AGREEMENT RELATING TO THIS MATERIAL, WHETHER OR NOT SUCH PARTY HAD ADVANCE NOTICE OF THE POSSIBILITY OF SUCH DAMAGES.

The material contained herein is not a license, either expressly or impliedly, to any IPR owned or controlled by any of the authors or developers of this material or MIPI. Any license to use this material is granted separately from this document. This material is protected by copyright laws, and may not be reproduced, republished, distributed, transmitted, displayed, broadcast or otherwise exploited in any manner without the express prior written permission of MIPI Alliance. MIPI, MIPI Alliance and the dotted rainbow arch and all related trademarks, service marks, tradenames, and other intellectual property are the exclusive property of MIPI Alliance Inc. and cannot be used without its express prior written permission. The use or implementation of this material may involve or require the use of intellectual property rights (“IPR”) including (but not limited to) patents, patent applications, or copyrights owned by one or more parties, whether or not members of MIPI. MIPI does not make any search or investigation for IPR, nor does MIPI require or request the disclosure of any IPR or claims of IPR as respects the contents of this material or otherwise.

Without limiting the generality of the disclaimers stated above, users of this material are further notified that MIPI: (a) does not evaluate, test or verify the accuracy, soundness or credibility of the contents of this material; (b) does not monitor or enforce compliance with the contents of this material; and (c) does not certify, test, or in any manner investigate products or services or any claims of compliance with MIPI specifications or related material.

Questions pertaining to this material, or the terms or conditions of its provision, should be addressed to:

MIPI Alliance, Inc.
c/o IEEE-ISTO
445 Hoes Lane, Piscataway New Jersey 08854, United States
Attn: Managing Director

**Special Note Concerning I3C and I3C Basic**

As described in the I3C Basic specification, certain parties have agreed to grant additional rights to I3C Basic implementers, beyond those rights granted under the MIPI Membership Agreement or MIPI Bylaws. Contribution to or other participation in the development of this FAQ document does not create any implication that a party has agreed to grant any additional rights in connection with I3C Basic. Consistent with the statements above, nothing in or about this FAQ document alters any party’s rights or obligations associated with I3C or I3C Basic.

# Contents

This page intentionally left blank.

# 1 Introduction

This FAQ has been developed to introduce the MIPI I3C ***[MIPI01]*** and I3C Basic ***[MIPI10]*** Specifications to developers and users. It answers questions raised to the MIPI Alliance Sensor Working Group (WG) after MIPI Board adoption of I3C v1.0, or late in development of v1.0 after the technical freeze by the WG and after MIPI Board adoption of I3C Basic v1.0.

The Sensor WG has compiled these frequently asked questions (FAQs) to assist Member implementation activity. Some areas also include clarification when an area of the Specification was ambiguous, and this FAQ will show the intended resolution of the ambiguity. Many of these topics reflect areas of planned improvements for a future update to I3C v1.0 and I3C Basic v1.0, showing how a v1.0 implementation can be better prepared and/or compliant in advance.

Throughout this FAQ document, unless otherwise noted, the terms ‘MIPI I3C’ and ‘I3C’ refer to both MIPI I3C ***[MIPI01]*** and MIPI I3C Basic ***[MIPI10]***.

***Note:***

> *None of the answers in this FAQ are intended to overwrite or overrule the information in the I3C Specification itself.*

The FAQ questions are organized into Sections by topic, based on the reader’s level of familiarity with the I3C Specification and implementation:

| Section | Title | Focus |
|---|---|---|
| 2.1 | Introduction to MIPI I3C | I’ve heard about I3C. Where can I read a bit more about it? |
| 2.2 | MIPI I3C Specification | I’ve started to read the I3C Specification. Tell me a bit more about the frequently asked questions, so I can better understand some of the details in the Specification. |
| 2.3 | Implementation: Ecosystem | Questions related to design kits, IP, test, and other parts of the enablement ecosystem. |
| 2.4 | Implementation: As a System Designer | Questions asked by early system designers. |
| 2.5 | Implementation: As a Software Developer | Questions asked by early software developers. |
| 2.6 | Interoperability Workshops | Questions asked by early Interoperability Workshop participants. |
| 2.7 | Up and Coming | Questions related to the next revision (and/or Errata) of the I3C Specification. |
| 2.8 | Clarification, Disambiguation, and Allowances for v1.0 | Questions related to ambiguous text in the I3C v1.0 and I3C Basic Specifications, and areas that a v1.0 device is allowed to enhance in advance of I3C v1.1. |
| 2.9 | Conformance Testing | Questions related to testing device conformance to the I3C Specification. |
| 2.10 | Legal & Intellectual Property Related Questions | Questions related to legal and IPR aspects of the I3C and I3C Basic Specifications and implementations. |

# 2 Frequently Asked Questions

This FAQ is organized into several topics:

- ***Section 2.1: Introduction to MIPI I3C***
- ***Section 2.2: MIPI I3C Specification***
- ***Section 2.3: Implementation: Ecosystem***
- ***Section 2.4: Implementation: As a System Designer***
- ***Section 2.5: Implementation: As a Software Developer***
- ***Section 2.6: Interoperability Workshops***
- ***Section 2.7: Up and Coming***
- ***Section 2.8: Clarification, Disambiguation, and Allowances for v1.0***
- ***Section 2.9: Conformance Testing***
- ***Section 2.10:Legal & Intellectual Property Related Questions***

## 2.1 Introduction to MIPI I3C

### Q1.1 What is MIPI I3C and I3C Basic?

MIPI I3C is a serial communication interface specification that improves upon the features, performance, and power use of I²C, while maintaining backward compatibility for most devices.

MIPI I3C Basic is technically identical to MIPI I3C, except with a reduced feature set and RAND-Z licensing (see ***Section 2.10***).

### Q1.2 What does the I3C acronym mean?

The official name is *MIPI Alliance Improved Inter Integrated Circuit*.

### Q1.3 Why is MIPI I3C being introduced?

The main purpose of MIPI I3C is threefold:

1. To standardize sensor communication,
2. To reduce the number of physical pins used in sensor system integration, and
3. To support low power, high speed, and other critical features that are currently covered by I²C and SPI.

MIPI I3C's purpose is now widening to cover many types of devices currently using I²C/SMbus, SPI, and UART.

### Q1.4 What are the main features of MIPI I3C?

MIPI I3C carries the advantages of I²C in simplicity, low pin count, easy board design, and multi-drop (vs. point to point), but provides the higher data rates, simpler pads, and lower power of SPI. I3C then adds higher throughput for a given frequency, in-band interrupts (from Slave to Master), dynamic addressing, advanced power management, and hot-join.

### Q1.5 How many signal lines does I3C have?

I3C has two signal lines: Data (SDA) and Clock (SCL).

### Q1.6 Does I3C require pull-up resistors on the bus like I²C?

No, I3C Masters control an active pull-up resistance on SDA, which they can enable and disable. This may be a board-level resistor controlled by a pin, or it may be internal to the Master.

### Q1.7 Who is I3C intended for?

I3C was initially intended for mobile applications as a single interface that can be used for all digitally interfaced sensors. However, it is now intended for all mid-speed embedded and deeply embedded applications across sensors, actuators, power regulators, MCUs, FPGAs, etc. The interface is useful for other applications, as it offers high-speed data transfer at very low power levels while allowing multi-drop, which is highly desirable for any embedded system.

### Q1.8 Why replace I$^2$C with I3C?

While I$^2$C has seen wide adoption over the years, it lacks some critical features – especially as mobile and mobile-influenced systems continue to integrate more and more sensors and other components. I$^2$C limitations worth mentioning include: 7-bit fixed address (no virtual addressing), no in-band interrupt (requires additional wires/pins), limited data-rate, and the ability of Slaves to stretch the clock (thus potentially hanging up the system, etc.). I3C aims both to fix these limitations and to add other enhancements.

### Q1.9 Does I3C use less power than I$^2$C?

The power consumption per bit transfer in all I3C modes is more efficient than I$^2$C, due to the use of push-pull (vs. open-drain) and strong pull-up signaling.

Further, I3C can save considerable device power through higher data rates (because the device can be put back to sleep sooner), built-in configuration and control (without intruding on the main communication protocols), in-band interrupt (IBI) as a low-cost wake mechanism, and the ability for Slaves to shut down all internal clocks while still operating correctly on the I3C bus.

### Q1.10 Is anyone currently using I3C?

I3C has been implemented and interoperability tested across a multitude of companies, and all of the major IP providers have products available. Silicon devices with I3C are becoming available, with even more coming throughout 2019.

### Q1.11 How is I3C different from I$^2$C?

I3C offers dynamic address assignment, Slave-initiated communication, and significantly higher communication speeds than I$^2$C.

### Q1.12 Is I3C backward compatible with I$^2$C?

Yes, most I$^2$C Slave devices can be operated on an I3C bus, as long as they have a 50 ns glitch filter and do not attempt to stall the clock. Such use will not degrade the speed of communications to I3C Slaves, requiring only decreased speed when communicating with the I$^2$C Slaves.

I3C Slave devices with a Static Address can operate as I$^2$C Slaves on an I$^2$C bus; optionally, they can also have a 50 ns spike filter..

### Q1.13 What is the maximum number of I3C devices that can be connected on the same bus?

The I3C bus is limited to around a dozen devices.

### Q1.14 Can there be more than one I3C Slave inside a chip?

Yes, multi-Slave I3C chips are possible.

### Q1.15 Can I3C and I$^2$C co-exist on the same bus?

Yes, both I3C and I$^2$C can share the same bus. See ***Question Q1.12*** regarding I3C backward compatibility for details.

### Q1.16 What is the bitrate for I3C?

I3C has several modes, each with associated bitrate(s). The base raw bitrate is 12.5 Mbps, with 11 Mbps real data rate at 12.5 MHz clock frequency (this is the only mode supported in I3C v1.0 and I3C Basic v1.0). The maximum raw bitrate is 33.3 Mbps at 12.5 Mhz, with real data rate of 30 Mbps (this is achieved via HDR modes that are currently only available in I3C v1.0).

Most traffic will use the 10 to 11 Mbps rate, while large messages can use one of the higher data rate modes.

### Q1.17 Can I3C Slaves initiate communication (i.e., interrupt the Master)?

Yes, I3C Slaves can initiate communication. Communication conflicts are solved by Slave address arbitration.

### Q1.18 Is it possible to have multiple Masters on the same I3C bus?

Yes, I3C allows for multiple Masters on the same bus. I3C has one Main Master that initially configures the bus and act as an initial current Master. Optionally, the bus can have multiple Secondary Master devices that initially act as Slaves. Any Secondary Master device can request to become the current Master. Once the current Master agrees, and transfers the current Master control to a given Secondary Master device, then that device becomes the current Master.

### Q1.19 How can Masters and Slaves communicate on the I3C bus?

The basic byte-based messaging schemes of I²C and SPI map easily onto I3C. Additionally, a set of common command codes (CCCs) has been defined for standard operations like enabling and disabling events, managing I3C specific features (dynamic addressing, timing control, etc.), and others. CCCs can be either broadcasted (sent to all devices on the bus), or directed at a specific device on the bus.

CCCs do not interfere with, and do not use up any of the message space of, normal Master-to -Slave communications. I3C provides a separate namespace for CCCs.

### Q1.20 What are CCCs (Common Command Codes) and why are they used?

The CCCs are the commands that an I3C Master uses to communicate to some or all of the Slaves on the I3C bus. The CCCs are sent to the I3C broadcast address (which is 7'h7E) so as not to interfere with normal messages sent to a Slave. The CCCs are used for standard operations like enabling/disabling events, managing I3C specific features and other bus operations. CCCs can be either broadcasted (sent to all the devices on the bus), or directed at specific devices on the bus. All CCCs (i.e., the command numbers) are allocated by MIPI Alliance, with some reserved for specific purposes that include MIPI Alliance enhancements and vendor extensions.

### Q1.21 Why replace SPI (Serial Peripheral Interface) with I3C?

SPI requires four wires and has many different implementations because there is no clearly defined standard. In addition SPI requires one additional chip select (or enable) wire for each additional device on the bus, which quickly becomes cost-prohibitive in terms of number of pins and wires, and power. I3C aims to fix that, as it uses only two wires and is well defined.

I3C covers most of the speed range of SPI, but is not intended for the highest speed grades that really only work well with a point-to-point interface, such as for SPI Flash.

## 2.2 MIPI I3C Specification

### Q2.1 How can the MIPI I3C Specifications be obtained?

The MIPI I3C Basic v1.0 Specification ***[MIPI10]*** was made publicly available in December 2018. Non-members may download the I3C Basic specification by visiting the MIPI I3C page on the MIPI Alliance website: https://www.mipi.org/specifications/i3c-sensor-specification. MIPI Alliance members have access and rights to the I3C Basic specification through their MIPI membership and member website.

MIPI Alliance members have access and rights to the MIPI I3C v1.0 Specification ***[MIPI01]*** through their MIPI membership and member website.

### Q2.2 Does I3C support inclusion of I$^2$C devices on the bus, and at what speed?

I3C supports legacy I$^2$C devices using Fast-mode (400KHz) and FastMode+ (1MHz) with the 50ns spike filter, but not the other I$^2$C modes, and not devices lacking the spike filter, or that stretch the clock.

### Q2.3 When is the pull-up resistor enabled?

Much of the activity on the I3C bus is in push-pull mode (that is, with the pull-up resistor disabled) in order to achieve higher data rate. However for some bus management activities, and for backwards compatibility with I$^2$C, pull-up-resistor-based open-drain mode is enabled. For example: arbitration during dynamic address assignment, and in-band interrupt. Also, the ACK/NACK during the 9th bit is done using pull-up resistor. With few exceptions, it is the responsibility of the I3C Master to provide open-drain class pull-up resistor when the bus is in the open drain mode.

### Q2.4 Is a High-Keeper needed for the I3C bus?

A high-keeper is used for Master-to-Slave and Slave-to-Master bus hand-off, as well as optionally when the bus is idle. The high-keeper may be a passive weak pull-up resistor on the bus, or an active weak pull-up or equivalent in the Master. The high-keeper only has to be strong enough to prevent system-leakage from pulling the bus low. At the same time, the high-keeper has to be weak enough that a Slave with a normal $I_{OL}$ driver is able to pull the bus line low within the minimum period.

### Q2.5 What is a Provisional ID, and why is it needed?

During bus initialization, the I3C Master assigns a 7-bit dynamic address to each device on the bus. For this to happen, each Slave device must have a 48-bit Provisional ID (that is, provisioned with its ID). The Provisional ID has multiple fields, including MIPI Manufacturer ID and a vendor-defined part number. The I3C Slave may also have a static address, which if the Master knows it, allows for faster assignment of the dynamic address.

### Q2.6 How do the first 32 bits of the Provisional ID work? Are they random or fixed?

The first part of the ID contains a unique manufacturer ID. Companies do not have to be MIPI Alliance members to be assigned a unique manufacturer ID.

The second part of the ID normally contains a part number (which is normally divided up into general and specific part info for that vendor), as well as possibly an instance number which allows for multiple instances of the same device on the same bus. The instance ID is usually fed from a pin-strap, or fuse(s), or non-volatile memory (NVM).

A random number may be used for the part number, although normally only for test mode, as set by the Master using the ENTTM (Enter Test Mode) CCC. When a device that supports random values enters the test mode, the PID[31:0] bits are randomized. When the Master exits the test mode, the devices reset bits PID[31:0] to their default value. The use of a random number allows for many instances of the same device to be attached to a gang programmer/tester, relying on the random number to uniquely give each a dynamic address.

### Q2.7 What if the Master detects a collision during Dynamic Address Assignment (DAA)?

With most configurations, this is not possible; each device will have its own manufacturer ID and part number, so no collisions are possible. If more than one instance of the same device is used on the same bus, then each instance must have a separate instance ID; otherwise there would be a collision. Likewise, if any device is using a random number for its part number, then multiple instances from that manufacturer could collide (i.e., could have the same random value that time).

If the Master knows the number of devices on the bus then it can detect this condition, since the number of dynamic addresses assigned would be less than the number of devices. Once detected, the I3C Master can take steps to resolve such collisions, for example by resetting DAA and restarting the process, or by declaring a system error after a set maximum number (e.g., 3) of such attempts fail.

### Q2.8 What CCCs must my Slave support before a dynamic address is assigned?

All I3C devices must be able to process broadcast CCCs at any time, whether or not they have been assigned a dynamic address (DA). For example, an I3C device may act as an I²C device before it gets its DA assigned. However, it is expected to ACK the START with 7’h7E; the only exception would be if this device choosing to remain only as an I²C device, in which case, it would leave any 50ns spike filter enabled. For those devices that do recognize START and address 7’h7E, those may see any CCC and not just ENTDAA (Enter DA Assignment) or SETDASA (Set Dynamic Address from Static Address). See the I3C Specification for the required CCCs. The device can determine the effect of each CCC based on whether or not it has the DA assigned (e.g., the RSTDAA CCC [Reset DA Address] will probably have no effect before the DA has been assigned).

### Q2.9 What are some of the I3C bus conditions when the bus is considered inactive?

In addition to open drain, pull-up and high-keeper, the I3C bus has three distinct conditions under which the bus is considered inactive: Bus Free, Bus Available, and Bus Idle.

- **Bus Free** condition is defined as a period occurring after a STOP and before a START and for a given duration (e.g., $t_{CAS}$ and $t_{BUF}$ timing).
- **Bus Available** condition is defined as a Bus Free condition for at least $t_{AVAL}$ duration. A Slave may only issue a START request (e.g., for In-Band Interrupt or Master Handoff) after a bus available condition.
- **Bus Idle** condition is defined to help ensure bus stability during hot-join events. This condition is defined as a period during which the Bus Available condition is sustained continuously for a duration of at least $t_{IDLE}$.

### Q2.10 When my device drives the bus, does it need to see a STOP before a Bus Idle?

For normal active I3C Slaves, yes. They should only drive the bus for an In-Band Interrupt (IBI) when they have seen a STOP and the $t_{BusAvailable}$ time has elapsed (about 1µs), and in response to a START (but not a Repeated START).

For hot-join devices, they do not know the bus condition, so they wait until the bus is IDLE, which means the SCL and SDA are both high for the $t_{IDLE}$ period (about 1ms).

### Q2.11 When can an I3C Slave issue an In-Band Interrupt (IBI)?

An I3C Slave can issue the IBI in the following two ways:

- Following a START (but not a Repeated START)
- If no START is forthcoming within the bus available condition, then an I3C Slave can issue a START request by pulling the SDA line low. The I3C Master would then complete the START condition by pulling the SCL clock line low and taking over the SDA.

### Q2.12 What is Hot-Join?

The I3C bus protocol supports a mechanism for Slaves to join the I3C bus after the bus is already configured. This mechanism is called Hot-Join. The I3C Specification defines the conditions under which a Slave can do that, e.g., a Slave must wait for a bus idle condition.

### Q2.13 Can I3C Hot-Join Slave devices be used on a legacy I$^2$C bus?

Only if they have a way to turn off the Hot-Join feature. Hot-Join is not compatible with I$^2$C masters, so Hot-Join would have to be disabled for the Slave to be used on a legacy I$^2$C bus. The disabling of the Hot-Join feature should be done via some feature that is not part of the I3C protocol (i.e., not via the DISEC command), since an I2C master does not support the I3C protocol.

### Q2.14 What are the I3C bus activity states?

The I3C bus activity states provides a mechanism for the Master to inform the Slaves about the expected upcoming levels of activity or inactivity on the bus, in order to help Slaves better manage their internal states (e.g., to save power).

The four activity states (and their expected activity interval) are:

- **Activity State 0:** Normal activity
- **Activity State 1:** Expect quiet for at least 100 µs
- **Activity State 2:** Expect quiet for at least 2 ms
- **Activity State 3:** Expect quiet for at least 50 ms

### Q2.15 Is there any time-stamping capability defined in the I3C bus?

***Note:***

*This question does not apply to I3C Basic v1.0.*

Yes. The I3C bus supports an optional time control mechanism. One mode is synchronous (from the synchronized timing reference) and four modes are asynchronous (Slave provides timestamp data). All I3C Masters are expected to support at least Async Mode 0.

- **Synchronous:** The Master emits a periodic time sync that allows Slaves to set their sampling time relative to this sync. This may be used in conjunction with one of the Asynchronous modes.
- **Asynchronous:** The Slaves apply their own timestamp to the data at the time they acquire samples, permitting the Master to time-correlate samples received from multiple different Slaves or sensors.

  There are four types of asynchronous time controls:

  - **Async Mode 0:** Basic mode that assumes that a Slave has access to a reasonably accurate and stable clock source to drive the time stamping – at least accurate for the duration of the time it has to measure (i.e., from event to IBI). A set of counters, in conjunction with IBI, are used to communicate time stamping information to the Master.
  - **Async Mode 1:** Advanced mode extends the basic mode by using some mutually identifiable bus events like, I3C START.
  - **Async Mode 2:** High precision mode that uses SCL falling edges (for SDR and HDR-DDR modes) as a common timing reference for Master and Slave. A burst oscillator is used to interpolate the time between a detected event and next SCL falling edge. For HDR-TSL and HDR-TSP modes, the mode uses both SDA transitions and SCL transitions as a timing reference.
  - **Async Mode 3:** Highest-precision triggerable mode that supports precise time triggering and measurement across multiple transducers applications like beam forming.

### Q2.16 Is there a maximum limit to I3C bus payload?

By default, there is no limit to the maximum message length. To reduce bus availability latency across multiple Slaves, the I3C bus allows negotiating for maximum message lengths between Master and Slave. Further, Read terminate is possible from the Master, to allow regaining control of the bus under a long message.

### Q2.17 Does the I3C bus enable ‘Bridges’?

Bridge devices are expected to enable an I3C bus to be bridged to other protocols, such as SPI, UART, etc. A CCC is defined to enable bridging devices, where the Master knows in advance that certain devices are bridges.

### Q2.18 Can a Slave indicate any speed limit that it might have?

All I3C Slaves must be tolerant of the 12.5 MHz maximum frequency, and all Slaves must be able to manage those speeds for CCCs. But Slaves may limit the maximum effective data rate for private message – either write, read, or both.

### Q2.19 Is there any test mode in the I3C bus?

Yes. A pair of directed and broadcast CCCs is available for the Master to enter/exit the test modes.

### Q2.20 Are there any error detection and recovery methods in I3C?

Yes, the I3C bus has elaborate error detection and recovery methods. Seven Slave error types (S0 to S6) and three Master error types (M0 to M2) are defined for the SDR mode, along with suggested recovery methods. Similarly, a set of errors are defined for each of the HDR modes.

### Q2.21 During HDR-DDR Mode CRC 5 transmission, how many clocks should I be looking for?

***Note:***

*This question does not apply to I3C Basic v1.0.*

The CRC transmission ends at bit 6 (counting down from 15), but bit 5 allows High-Z.

### Q2.22 Can a Master issue a STOP condition regardless of whether or not a Slave has issued an acknowledgment indicating a completed transaction?

The STOP can be issued anywhere the Slave is not driving the SDA during SCL high. It may not be appropriate to do so in terms of completion of a message. But ACK and completed transaction do not belong together in I3C.

## 2.3 Implementation: Ecosystem

### Q3.1 Who is defining the MIPI I3C specifications?

The I3C specification is defined by the MIPI Sensor Working Group formed in 2013, and I3C Basic is defined by the I3C Basic Ad-Hoc Working Group formed in 2018. Both groups are MIPI Alliance initiatives.

### Q3.2 Which companies are part of the MIPI Sensor Working Group?

The MIPI Sensor Working Group includes representatives from AMD, Broadcom, Cadence, IDT, Intel, InvenSense, Knowles, Lattice Semiconductor, MediaTek, Mentor Graphics, Nvidia, NXP, Qualcomm, QuickLogic, Sony, STMicroelectronics, Synopsys, VLSI Plus, and others. It is chaired by Ken Foust of Intel, and vice-chaired by Satwant Singh of Lattice Semiconductor.

### Q3.3 What is the availability of development hardware for I3C prototyping, including FPGAs?

A few vendors have provided FPGA based design kits, including some low-cost FPGAs that might be good enough for smaller production runs.

### Q3.4 What is the I3C IP core availability in the market?

Some vendors have started to offer Slave and/or Master IP cores for integration into ASIC devices and FPGAs, including a free-of-cost Slave IP available for prototyping and integration.

## 2.4 Implementation: As a System Designer

### Q4.1 What is the maximum capacitance load allowed on the I3C bus?

The I3C Specification lists the maximum per-device capacitance on SCL and SDA, but the goal is that most or all devices will be well below that. Capacitance alone is not sufficient to determine maximum frequency on the I3C bus (as with any bus). It is important to consider maximum propagation length, effect of stubs, internal clock-to-data ($t_{SCO}$) of the Slaves, as well as capacitive load.

### Q4.2 What is the maximum wire length for I3C communication?

The maximum wire length would be a function of speed, as all the reflections and bus turnaround must complete within one cycle. Larger distances can be achieved at the lower speeds than at the higher ones. For example at 1 meter (between Master and Slave), the maximum effective speed is around 6 MHz for read, to allow for clock propagation time to Slave and SDA return time to Master.

### Q4.3 Can I$^2$C repeaters be used for I3C?

Not directly, for a couple of reasons:

1. The I3C bus works with push-pull modes (in addition to the open drain for some transfers), and
2. Much higher speeds. Most such devices are quite limited in speed, because of the lag effect of changing states on SCL and SDA due to both series-resistance and assumptions about open-drain.

Long wire approaches are being evaluated for a future version of the I3C Specification.

### Q4.4 Will the I$^2$C devices respond to I3C commands?

No. The I3C CCCs are always preceded by I3C broadcast address 7’h7E. Since the I$^2$C specification reserves address 7’h7E, no legacy I$^2$C Slave will match the I3C broadcast address, and thus would not respond to the I3C commands. Likewise, the dynamic address assigned to I3C devices would not overlap the I$^2$C static addresses, so they would not respond to any I3C address (even if they could see it).

### Q4.5 How are communication conflicts resolved on the I3C bus?

The I3C Slaves are only allowed to drive the bus under certain situations. Besides during a read and when ACKing their own address, they may also drive after a START (but not Repeated START). After a START, the I3C bus reverts back to open-drain pull-up resistor mode, thus the Slave that drives a low value (logic 0) would win.

### Q4.6 Can I3C devices cause the communication bus to ‘hang’?

Unlike I$^2$C, there is no natural way to ‘hang’ the bus. In I$^2$C, clock stretching (where the Slave holds the clock low, stopping it from operating) often causes serious problems with no fix: there’s simply no way to get the Slave’s attention if it has hung the bus. By contrast, in I3C only the Master drives the clock, and so the Slave performs all actions on SDA relative to that clock, thereby eliminating the normal causes of such hangs.

Further, since I3C is designed to ensure that I3C Slaves can operate their back-end I3C peripheral off the SCL clock (vs. oversampling), problems elsewhere in the Slave will not translate into bus hangs.

If a system implementer is highly concerned about a Slave accidently locking itself, then a separate hard-reset line could be used. For the next revision of the I3C and I3C Basic Specifications, a feature called Slave Reset is being added to reset non-responsive I3C Slaves (i.e., if a Master emits a certain pattern that does not occur during regular communication, then the devices on the bus would treat it just like a hardwired reset line).

### Q4.7 Will all I3C devices be compatible with all CCCs?

No. Some CCCs are mandatory, while others are optional and a given device will either support them or not, depending upon the device's capabilities.

## 2.5 Implementation: As a Software Developer

### Q5.1 Are there any companion MIPI I3C specifications that enable SW development?

Yes. The following MIPI Specifications are expected to help with SW development:

- *MIPI Specification for I3C Host Controller Interface (I3C HCI), v1.0* ***[MIPI02]***

  Creates a standard definition that allows a single OS driver (aka 'in-box driver') to support I3C hardware from several vendors, while also allowing vendor-specific extensions or improvements. The target audience of the HCI Specification is application processor host controllers; in particular, developers of host controller (i.e., I3C main Master) hardware, and developers of I3C host controller software.
- *MIPI Specification for Discovery and Configuration (DisCo), v1.0* ***[MIPI03]***

  Describes a standardized device discovery and configuration mechanism for interfaces based on MIPI Specifications, which can simplify component design and system integration. Also oriented to application processors.
- *MIPI DisCo Specification for I3C, v1.0* ***[MIPI04]*** (In development)

  Allows operating system software to use ACPI (Advanced Configuration and Power Interface) structures to discover and configure the I3C host controller and attached devices in ACPI-compliant systems. Also oriented to application processors.
- In addition, a *System Integrators Application Note for I3C v1.0 and I3C Basic v1.0, v1.0* ***[MIPI05]*** has been developed to help ASIC hardware developers, system designers, and others working in the more deeply embedded I3C devices.

### Q5.2 Are there software libraries available (or about to be) for I3C?

Core I3C infrastructure is being added to the Linux Kernel via patchwork.kernel.org.

## 2.6 Interoperability Workshops

### Q6.1 What is a MIPI I3C Interoperability Workshop?

It is a MIPI Alliance sponsored event where different vendors bring their I3C implementations and check interoperation with other vendors.

### Q6.2 What is the output from a MIPI I3C Interoperability Workshop?

There are three major outputs from a MIPI I3C Interoperability Workshop:

- The vendors can get detailed information about how well their I3C implementations interoperate with other vendors' implementation. Vendors can also compare their results with one another.
- MIPI Alliance can generate an overall picture that shows the industry state-of-the-I3C-implementaion. For example, how many vendors have implemented I3C, and how many implementations pass or fail against one another.
- The MIPI Sensor Work Group (SWG) gets better understanding about any major issues with the I3C Specification. The SWG can then leverage that learning by adding to this FAQ, the *System Integrators Application Note for I3C v1.0 and I3C Basic v1.0* ***[MIPI05]***, and the next revision of the I3C Specification.

### Q6.3 Are MIPI I3C Interoperability Workshops an ongoing activity?

MIPI arranges a particular I3C Interoperability Workshop event in response to requests from its membership.

### Q6.4 Who can attend or participate in a MIPI I3C Interoperability Workshop?

In general, any MIPI Alliance members who have I3C hardware ready to interop can participate.

### Q6.5 What HW/SW is typically needed to participate in a MIPI I3C Interoperability Workshop?

While this could change in future, the minimal requirements to date have been the availability of a board with an I3C device that can connect to other devices via the three wires SDA, SCL, and GND. It's also useful to have software (e.g., running on a laptop connected to the board and I3C device) to interactively view transmitted and received bus communications, but this might not be required for Slaves.

Currently there are solutions working at 3.3V and 1.8V.

## 2.7 Up and Coming

### Q7.1 What future MIPI specifications will be leveraging I3C?

Many other MIPI Working Groups are in the process of leveraging the I3C specification. As of the writing of this FAQ, the list includes:

- **Camera WG:** Camera Control Interface (CCI) chapter of the *MIPI Specification for Camera Serial Interface 2 (CSI-2)*, v2.1 ***[MIPI06]***
- **Display WG:** Multiple MIPI Specifications for Touch interfaces
- **Debug WG:** *MIPI Specification for Debug for I3C* ***[MIPI07]*** (In development)
- **RIO WG** (Reduced I/O)**:** *MIPI Specification for Virtual GPIO Interface (VGI$^{SM}$)* ***[MIPI08]***, reducing number of GPIOs used via I3C (In development)

### Q7.2 Are there any impending MIPI I3C fixes/errata that should be applied now?

Based on learning from the early implementations, I3C Interoperability Workshops, queries from adopters, and reviews by the Sensor WG, this FAQ represents clarifications, planned improvements that can be implemented by I3C v1.0 devices, and other hints about what's coming next.

### Q7.3 Are any revisions to MIPI I3C v1.0 expected?

Currently, the MIPI Sensor WG is working towards Version v1.1 of the I3C and I3C Basic Specifications, which is expected to add incremental new features while staying backwards compatible with v1.0. That is, v1.1 features which do not impact v1.0 devices will simply be available, whereas others will be enabled based on the capabilities of various Slaves on a v1.1 bus, just as is done with optional features in v1.0.

### Q7.4 What new features, if any, are coming to MIPI I3C or I3C Basic?

While not finalized yet, the following features are under consideration:

- Grouped addressing
- Additional Slave error detection/recovery
- CCC support in HDR-DDR/TSP
- HDR-DDR end write
- HDR-TSP end transfer
- Clock-to-Data refinement
- Timing Control disable
- New minimum $t_{IDLE}$
- Slave Reset
- Multi-Lane, for speed
- HDR-DDR-end CRC
- HDR-BT (Bulk Transport Mode)

## 2.8 Clarification, Disambiguation, and Allowances for v1.0

### Q8.1 Can Sync and Async time control be enabled at the same time?

***Note:***

*This question does not apply to I3C Basic v1.0.*

Yes. This is allowed such that the ODR (Output Data Rate) rate controls the IBI rate, and the Async Mode timestamp on the IBI indicates how long ago the sample was collected.

### Q8.2 Is there a way to turn off Time Control?

***Note:***

*This question does not apply to I3C Basic v1.0.*

Yes, the SETXTIME code of 0xFF may be used. The MIPI Sensor WG plans to officially define this in I3C v1.1, but I3C v1.0 devices can support it now.

### Q8.3 Do I have to wait the full 1ms for Hot-Join?

***Note:***

*This question does not apply to I3C Basic v1.0.*

In I3C v1.1 the MIPI Sensor WG plans to officially define a new $t_{IDLE}$ minimum value of 200 µs, since that is sufficient for all valid uses. I3C v1.0 devices may choose to support that smaller delay now. I3C Basic v1.0 already supports a $t_{IDLE}$ minimum value of 200 µs.

### Q8.4 Is there any way to exit from an S0/S1 error other than waiting for an Exit Pattern?

Yes. An I3C Slave may watch for 60 µs with no SCL or SDA changes to make sure that the bus is not in HDR mode (and therefore must be in SDR mode). After that, it is appropriate to wait for START (assumed to be Repeated START) or STOP.

### Q8.5 What happens if the Master crashes during a Read?

The I3C Slave may choose to detect 100 µs without an SCL edge. If that happens, it can abandon the Read and release SDA to avoid a bus hang when the Master restarts.

### Q8.6 If a device has a $t_{SCO}$ value greater than 12 ns, does that mean it doesn't qualify as an I3C Device?

***Note:***

*This question does not apply to I3C Basic v1.0.*

No. The $t_{SCO}$ (Clock to Data Turnaround delay time) is information provided by Slaves so that system designers can properly compute the maximum effective frequency for reads on the bus. The $t_{SCO}$ number is meant to be used together with the line capacitance (trace length) and number of Slaves and stubs (if present).

However, I3C Slaves with $t_{SCO}$ delay greater than 12 ns must:

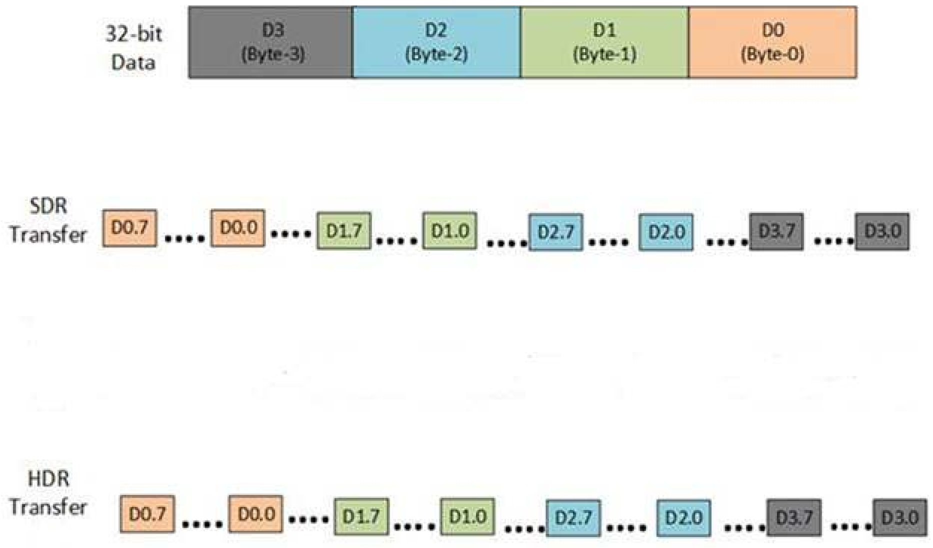
- Set the Limitation bit in the Bus Characteristics Register (BCR) to 1'b1,
- Set the Clock-to-Data Turnaround field of the maxRD Byte to 3'b111, and
- Communicate the $t_{SCO}$ value to the Master by private agreement (i.e., product datasheet).
- I3C Basic v1.0 already clarifies this.

### Q8.7 Does Figure 44 HDR-DDR Format apply for Command, Data, and CRC? Or only for Data?

***Note:***

*This question does not apply to I3C Basic v1.0.*

Only for Data. The Data bytes are sent the same way as in SDR Mode. The following figure shows how Data is transferred from memory to the I3C Data Line:



The Command and CRC Byte are each transferred as a packet instead:

**Command** (MSb to LSb):

- Preamble (2 bits)
- Command (8 bits)
- Slave address (7 bits)
- Reserved bit (1 bit)

**CRC** (MSb to LSb):

- Preamble (2 bits)
- Token (4 bits)
- CRC Byte (5 bits)
- Setup bits (2 bits)

### Q8.8 Does the Master provide an additional CLK after the Terminate Condition and before the Restart/Exit Pattern, as shown in Figure 52 Master Terminates Read?

***Note:***

*This question does not apply to I3C Basic v1.0.*

No, there is an error in Figure 52. The beginning of the Restart/Exit Pattern should show SCL Low and SDA changing.

### Q8.9 In a Hot-Join, when should the DISEC CCC be sent? After ACK, or after NACK?

The Hot-Join mechanism allows the Master to first NACK, and then send the DISEC CCC to stop the Hot-Join. If the Master ACKs, that is interpreted as a promise that it will eventually send the ENTDAA CCC.

## 2.9 Conformance Testing

### Q9.1 What is a MIPI Conformance Test Suite (CTS)?

A MIPI WG develops a Conformance Test Suite (CTS) document in order to improve the interoperability of products that implement a given MIPI interface Specification. The CTS defines a set of conformance or interoperability tests whereby a product can be tested against other implementations of the same Specification.

### Q9.2 Is there a CTS for I3C v1.0?

Yes. A CTS for I3C v1.0 is being drafted by the MIPI Alliance Sensor WG ***[MIPI09]***.

### Q9.3 What is the scope of tests for the I3C v1.0 CTS?

The CTS tests are designed to determine whether a given product conforms to a subset of the requirements defined in the I3C Specification v1.0. The scope of this first version of the CTS is intentionally limited, in order to meet time-to-market requirements imposed by the rapid adoption of I3C in the marketplace, focusing on:

1. SDR-only Devices without optional I3C capabilities,
2. All Master and Slave Error Detection and Recovery methods, and
3. Basic HDR Enter/tolerance/Restart/Exit are in scope, but HDR-DDR is under consideration.

Considering the CTS a living document, the Sensor WG plans to continue expanding the scope of the CTS through future revisions that eventually encompass all required and optional features of the I3C Specification.

The CTS tests are organized as Master device under test (DUT) and Slave DUT. Tests for each are presented in the order in which they appear in the I3C Specification, to simplify identification of pertinent detail between the two documents.

### Q9.4 Does the I3C Interoperability Workshop follow the I3C CTS?

Interoperability Workshops will ultimately follow the tests identified in the I3C CTS, as it nears completion.

### Q9.5 What details are provided for each I3C CTS test case?

Each test in the I3C CTS will contain:

- A clear purpose
- References
- Resource requirements
- Tracked last technical modification
- Discussion
- All test case detail (i.e., setup, procedure, results, problems, and MIPI Product Registry inclusion).

DC/AC parametric requirements will be embedded in each test, not split out into a separate PHY-related CTS or subsection.

## 2.10 Legal & Intellectual Property Related Questions

### Q10.1 Is MIPI I3C Basic royalty free?

The parties that directly developed the MIPI I3C Basic specification have agreed to license all implementers on royalty free terms, as further described in the I3C Basic specification document ***[MIPI10]***. Further, all implementers of the I3C Basic specification must commit to grant a reciprocal royalty free license to all other implementers if they wish to benefit from these royalty free license commitments. And, of course, MIPI itself does not charge royalties in connection with its specifications. MIPI's intent is to create a robust royalty free environment for all implementers of I3C Basic.

No set of IPR terms can comprehensively address all potential risks, however. The terms apply only to those parties that agree to them, for example, and the scope of application is limited to what is described in the terms. Implementers must ultimately make their own risk assessment.

### Q10.2 What license terms apply to MIPI I3C v1.x?

MIPI's regular IPR terms apply to the full MIPI I3C specification. MIPI's terms require that members make licenses available only to other members, as described in the MIPI Membership Agreement and MIPI Bylaws. To benefit from the license commitments, a party must be a MIPI member.

For features that are included in MIPI I3C Basic, a MIPI member can implement under the regular IPR terms, or can opt to implement the feature under the I3C Basic framework. If a member opts in to the I3C Basic framework, then they must grant the reciprocal licenses required under that framework. MIPI Alliance members are not required to participate in the I3C Basic license framework, however. Features of I3C 1.x that are not included in I3C Basic are subject only to MIPI's regular IPR terms.

Prior to the release of I3C Basic, MIPI had made certain versions of the full MIPI I3C specification available for public review, under "copyright only" terms – that is, MIPI published the specification, but noted that no rights to implement the specification were granted under any party's patent rights. MIPI no longer publishes the full I3C specification publicly. A non-member is not granted any right to implement the full MIPI I3C 1.x specification, either by MIPI or any MIPI member.

I3C Basic is available to non-members, as described in question ***Q2.1 How can the MIPI I3C Specifications be obtained?***.

# 3 Terminology

See also ***Section 2*** in the MIPI I3C v1.0 Specification ***[MIPI01]***.

## 3.1 Definitions

**ACK:** Short for “acknowledge” (an I3C bus operation)

**Bus Available:** I3C bus condition in which a device is able to initiate a transaction on the bus.

**Bus Free:** I3C bus condition after a STOP and before a START with a duration of at least $t_{CAS}$.

**Bus Idle:** An extended duration of the Bus Free condition in which devices may attempt to Hot-Join the I3C Bus.

**High-Keeper:** A weak Pull-Up type Device used when SDA, and sometimes SCL, is in High-Z with respect to all Devices.

**Hot-Join:** Slaves that join the I3C bus after it is already started, whether because they were not powered previously or because they were physically inserted into the Bus. The Hot-Join mechanism allows the Slave to notify the Master that it is ready to get a dynamic address.

**In-Band Interrupt (IBI):** A method whereby a Slave device emits its Address into the arbitrated Address header on the I3C bus to notify the Master of an interrupt.

**Main Master:** Master that has overall control of the I3C bus.

**Master:** The I3C bus device that is controlling the bus.

**Slave:** An I3C Slave device can only respond to either Common or individual commands from a Master. A Slave device cannot generate a clock.

## 3.2 Abbreviations

| | |
|---|---|
| DisCo | Discovery and Configuration (family of MIPI Alliance interface Specifications) |
| e.g. | For example (Latin: exempli gratia) |
| i.e. | That is (Latin: id est) |

## 3.3 Acronyms

See also the acronyms defined in the MIPI I3C v1.0 Specification ***[MIPI01]***.

| | |
|---|---|
| CCC | Common Command Code (an I3C common command or its unique code number) |
| CTS | Conformance Test Suite |
| DAA | Dynamic Address Assignment (an I3C bus operation) |
| FAQ | Frequently Asked Questions |
| HCI | Host Controller Interface (a MIPI Alliance interface Specification ***[MIPI02]***) |
| HDR | High Data Rate (a set of I3C bus modes) |
| HDR-DDR | HDR Double Data Rate (an I3C bus mode) |
| HDR-TSP | HDR Ternary Symbol for Pure Bus (an I3C bus mode) |
| I3C | Improved Inter Integrated Circuit (a MIPI Alliance interface Specification ***[MIPI01]***) |
| IBHR | In-Band Hardware Reset (an expected I3C bus feature) |
| IBI | In-Band Interrupt (an I3C bus feature) |
| ODR | Output Data Rate |
| SCL | Serial Clock (an I3C bus line) |
| SDA | Serial Data (an I3C bus line) |
| SDR | Single Data Rate (an I3C bus mode) |
| SPI | Serial Peripheral Interface (an interface specification) |

# 4 References

[MIPI01] *MIPI Alliance Specification for I3C® (Improved Inter Integrated Circuit)*, version 1.0, MIPI Alliance, Inc., 23 December 2016 (Adopted 31 December 2016).

[MIPI02] *MIPI Alliance Specification for I3C Host Controller Interface (I3C HCI℠)*, version 1.0, MIPI Alliance, Inc., 29 September 2017 (Adopted 4 April 2018).

[MIPI03] *MIPI Alliance Specification for Discovery and Configuration (DisCo℠)*, version 1.0, MIPI Alliance, Inc., 1 July 2016 (Adopted 28 December 2016).

[MIPI04] *MIPI Alliance DisCo℠ Specification for I3C℠*, version 1.0, MIPI Alliance, Inc., In press.

[MIPI05] *MIPI Alliance System Integrators Application Note for I3C® v1.0 and I3C Basic ℠ v1.0*, App Note version 1.0, MIPI Alliance, Inc., 8 December 2018 (approved 8 December 2018).

[MIPI06] *MIPI Alliance Specification for Camera Serial Interface 2 (CSI-2℠*), version 2.1, MIPI Alliance, Inc., 14 December 2017 (Adopted 9 April 2018).

[MIPI07] *MIPI Alliance Specification for Debug for I3C℠*, version 1.0, MIPI Alliance, Inc., In press.

[MIPI08] *MIPI Alliance Specification for Virtual GPIO Interface (VGI℠*), version 1.0, MIPI Alliance, Inc., In press.

[MIPI09] *MIPI Alliance Conformance Test Suite (CTS) for I3C℠ v1.0*, CTS version 1.0, MIPI Alliance, Inc., In press.

[MIPI10] *MIPI Alliance Specification for I3C Basic℠ (Improved Inter Integrated Circuit)*, version 1.0, MIPI Alliance, Inc., 19 July 2018 (Adopted 8 October 2018).

This page intentionally left blank.