I3C® Application Note: General Topics

Applies to MIPI I3C v1.1+ and MIPI I3C Basic v1.1.1+

App Note Version 1.1
27 April 2022

MIPI Board Approved 27 July 2022
Public Release Edition

This is an informative document, not a MIPI Specification. MIPI member companies' rights and obligations apply to this Supporting Document as defined in the MIPI Membership Agreement and MIPI Bylaws.

Various rights and obligations that apply solely to MIPI Specifications (as defined in the MIPI Membership Agreement and MIPI Bylaws) including, but not limited to, patent license rights and obligations, do not apply to this document.

This document is subject to further editorial and technical development.

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 App Note 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 App Note document alters any party's rights or obligations associated with I3C or I3C Basic.

Contents

Figures

Release History

Date	Version	Description
08-Dec-2018	v1.0	Initial Board approved release.
27-Jul-2022	v1.1	Board approved release.

1 Introduction

The MIPI I3C Bus interface ***[MIPI06]*** is an evolutionary specification that improves upon the legacy I²C standard. It is designed to reduce the number of physical pins used in sensor system integration, and supports low-power, high-speed digital communication typically associated with UART and SPI interfaces.

Note:

*When the term "I3C Specification" is used in reference to **[MIPI06]** in this Application Note, it means the current state of I3C in a generic sense (i.e., the most recently adopted version of either the I3C Specification or the I3C Basic Specification).*

I3C's main features include:

- Support for lower voltage: 1.2V for typical applications, and down to 1.0V in I3C Basic
- Dynamic Addressing assignment
- High Data Rate (HDR) Modes with reduced energy requirements
- Multi-Controller and Multi-Drop capabilities
- In-Band Interrupts
- Hot-Join support
- Backward compatibility with I²C

The I3C interface plays a fundamental role in streamlining sensor integration in smartphones, wearables, and Internet-of-Things (IoT) devices. The I3C interface is also extensible to newer and more advanced use cases that go beyond the original scope of sensor integrations.

This Application Note is intended to help users understand how the I3C interface works, by presenting a range of topics that are relevant for platform architects, HW designers, system integrators, SW developers, and other engineers who enable and support systems that use I3C Buses.

1.1 Scope

This General Topics Application Note is intended to guide several different groups:

- Those developing MIPI I3C Controller and Target Devices to understand how their parts will fit into different types of systems, and the considerations for functionality and features.
- System Designers who need to design systems that integrate such I3C Devices, and potentially Legacy I²C Devices, and who need to know considerations of trace layout and connections, voltage regulation for the Devices, any strapping or other ID factors, etc.
- MIPI I3C Controller SW developers, including those who must pay special consideration to systems with multiple I3C Controllers on the same I3C Bus. This includes users of both standardized Host Controller APIs and MCU/DSP firmware.

This Application Note has several parts, each focusing on a different area and covering both required considerations and optional ones, based on which features and topology are used in a given system. This approach makes it easier for any of the targeted groups to focus on what matters to them based on what configurations they will be working with.

This Application Note is intended to be used together with the latest I3C Specification ***[MIPI06]***. Each Application Note section corresponds to one or more Specification sections, primarily focusing on Specification ***Section 6, I3C Electrical Specifications***. The Application Note sections amplify the Specification with additional context (e.g., analysis data to back up recommended use models) and details (e.g., total Bus capacitance vs. per-Device capacitance allowances) that might not be presented in the protocol specification itself.

2 Terminology

See also ***Section 2*** in the MIPI I3C Specification ***[MIPI06]***.

2.1 Definitions

System Designer: Engineer designing a system that includes an I3C Bus.

2.2 Abbreviations

e.g.	For example (Latin: exempli gratia)
i.e.	That is (Latin: id est)

2.3 Acronyms

CCC	Common Command Code
HDR	High Data Rate
HJ	Hot-Join
I3C	MIPI Improved Inter Integrated Circuit interface, or its Specification document ***[MIPI06]***
IBI	In-Band Interrupt
SDR	Single Data Rate

3 References

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

[MIPI02] *MIPI Alliance Specification for I3C (Improved Inter Integrated Circuit)*, version 1.1 incorporating Errata 01, MIPI Alliance, Inc., 27 December 2019 (MIPI Board Adopted 11 December 2019); Errata 01 approved 24 June 2020.

[MIPI03] *MIPI Alliance Specification for I3C (Improved Inter Integrated Circuit)*, version 1.1.1, MIPI Alliance, Inc., 11 June 2021 (MIPI Board Adopted 8 June 2021).

[MIPI04] *MIPI Alliance Specification for I3C Basic℠ (Improved Inter Integrated Circuit – Basic)*, version 1.0, MIPI Alliance, Inc., 19 July 2018 (MIPI Board Adopted 8 October 2018).

[MIPI05] *MIPI Alliance Specification for I3C Basic℠ (Improved Inter Integrated Circuit – Basic)*, version 1.1.1, MIPI Alliance, Inc., 9 June 2021 (MIPI Board Adopted 23 July 2021).

[MIPI06] Either ***[MIPI03]*** or ***[MIPI05]***.

Note:

When the term "I3C Specification" is used in reference to ***[MIPI06]****, it means the current state of I3C in a generic sense (i.e., the most recently adopted version of either the I3C Specification or the I3C Basic Specification).*

[MIPI07] MIPI Alliance, Inc., "Current I3C Device Characteristic Register (DCR) Assignments", <https://www.mipi.org/MIPI_I3C_device_characteristics_register>, last accessed 27 July 2022.

[MIPI08] *MIPI Alliance I3C Application Note: Virtual Devices and Virtual Targets*, App Note version 1.0, MIPI Alliance, Inc., 30 August 2021 (MIPI Board Approved 4 September 2021).

[MIPI09] *MIPI Alliance I3C Application Note: Hot-Join*, App Note version 1.0, MIPI Alliance, Inc., 30 August 2021 (MIPI Board Approved 4 September 2021).

[MIPI10] *MIPI Alliance Conformance Test Suite for I3C v1.1.1 and I3C Basic v1.1.1*, CTS version 1.0, MIPI Alliance, Inc., 4 August 2021 (MIPI Board Approved 5 August 2021).

[NXP01] UM10204, *I2C-bus specification and user manual*, Rev. 7.0, NXP Semiconductors, 1 October 2021.

This page intentionally left blank.

4 Overview

This Application Note describes two broad categories of information:

- Details of a few typical and supported topologies, including different trace distances. This covers the allowances and challenges presented by each topology.
- Details of added functionality and allowances, including Hot-Join and Hot-insertion, Timing control (timestamping) considerations, power states of Devices and the implications for the Bus, etc.

4.1 Supported Topologies

This Application Note provides additional guidance not found in the I3C Specification itself ***[MIPI06]***. System Designers who are developing I3C-based systems are encouraged to consult the sections that are relevant to their designs.

Analysis data is provided to help understand the considerations that impact the placement and usage of mixed Devices. This analysis data can also be extrapolated to other configurations. It could be used, for example, when choosing what types of Devices to mix on one Bus with a given topology (e.g., single Bus configuration vs. two-Bus configuration), when deciding how best to manage relative Device locations (floor-planning), and when gauging system reliability.

4.2 Using Added Functionality

The I3C Specification ***[MIPI06]*** includes optional advanced capabilities and features, giving the System Designer the flexibility to choose which ones to use for a particular implementation. Many of these features allow for efficient implementations that can be applied to diverse use cases. The choice of added features may have implications for the System Designer, and for software on the Controller (i.e., on the Host).

The I3C Specification provides an exact and detailed description of how each feature works with the I3C protocol. The specification is quite extensive, as it includes numerous possible configurations and special cases, and as a result can be challenging for new readers to absorb. By contrast, this Application Note is more descriptive and provides additional information on how the features can be incorporated into a system. System Designers can use this additional information for easier, faster assessment of whether the features are applicable for their designs.

This page intentionally left blank.

5 System Integration Guidelines

This section discusses key guidelines that system integrators will want to follow in their designs, with the goal of optimizing an I3C Bus for various topologies. These are intended either for Pure Bus configurations (i.e., Buses with only I3C Devices), or for Mixed Bus configurations (Buses with both I3C Devices and Legacy I^2C Devices).

The guidelines focus on:

- I3C Bus Operation
- I3C Device Characteristics
- Legacy I^2C Device Characteristics
- Dynamic Address Assignment
- Bus Topologies
- Physical/Electrical/Testing Considerations (CTS)
- Bus High Keeper feature

5.1 I3C Bus Operation

An I3C Bus operates with an active Controller and one or more Targets, exchanging data using two lines, SDA and SCL, with many similarities with the I^2C bus ***[NXP01]***. Controllers and Targets have a number of types and roles as detailed in ***Section 5.2.1*** of this Application Note, ***Devices Roles and Responsibilities***.

The Bus begins operation in the default SDR Mode (Single Data Rate, see specification ***Section 5.1 [MIPI06]***) where data clocking shares many similarities with I^2C. During a Bus Transaction, the Bus may switch into one of the HDR Modes described later.

In SDR Mode, the meaning and operation of the SDA and SCL lines are:

- **SCL:** The **S**erial **CL**ock line is the used by the Controller to clock data on the SDA line. SCL is driven by the Active Controller in Push-Pull mode, with a typical 4 mA drive strength, which results a reasonable transition time for a Bus configuration up to 50 pF Bus capacitance.
- **SDA:** The **S**erial **DA**ta line mostly carries data in/out, and is also used for additional signaling purposes. Unlike SCL, SDA can be driven either by the Active Controller or by a Target. SDA can operate in either Open Drain or Push-Pull configuration, depending on the Bus state during Bus transactions.

During the SDR data shifting operation, SDA can change state only when SCL is low. The bit value is latched following the SCL rising edge; see ***Figure 235*** and ***Figure 236*** from the I3C Specification ***[MIPI03]*** (***Figure 146*** and ***Figure 147*** from the I3C Basic Specification ***[MIPI05]***) for data set-up and holding times. The SDA/SCL line operation changes when the Bus enters into one of the HDR Modes (High Data Rate) where SCL/SDA are operated differently, as explained below in ***Section 5.1.4*** of this Application Note, ***High Data Rates (HDR Modes)***.

I3C Bus transactions are delimited by the same START (S), Repeated START (Sr), and STOP (P) conditions that I^2C uses.

The timing specifications are defined for three different operating conditions:

- **I3C is in Open Drain state** per ***Table 122*** in the I3C Specification (***Table 86*** in the I3C Basic Specification)
- **I3C is in Push-Pull state**, covering the SDR, HDR-DDR, and HDR-BT Modes per ***Table 123*** in the I3C Specification (***Table 87*** in the I3C Basic Specification) and HDR-TSP and HDR-TSL Modes as per ***Table 124*** in the I3C Specification.
- **I3C is communicating with I^2C Legacy Device(s)**, per ***Table 121*** in the I3C Specification (***Table 85*** in the I3C Basic Specification)

Care must be taken because most of the timing parameters, such as rise/fall time, data set-up, etc., will be different for the three operating conditions that will change dynamically.

5.1.1 Format of Data Transfer Units

A Bus transmission is done by sending 9-bit sequences. A sequence can be either Control or Data. One or more Control sequences are sent first, then Data sequences may follow.

- **Control Sequence format:**
  - Either a **7-bit Address** or a **Command Code**, issued by the Controller.
  - **RnW bit:** 1=Read, 0=Write, issued by the Controller.
  - **ACK bit:** 0=ACK, 1=NACK (i.e., not-ACK). Emitted by one or more Targets.

    Note:

    During the ACK period, the SDA is always put in Open Drain to allow multiple targets to pull SDA low to signal the ACK. In some special cases (for example, In-Band Interrupt and Hot-Join), the Active Controller may also ACK.
- **Data Sequence:**
  - **8-bit Data:** Read or Write, depending on the RnW bit of the previous Control Sequence.
  - **9th bit:** The meaning of this bit differs between Read transfers and Write transfers:
    - **During Writes** the 9th bit indicates the parity of the Data, as an integrity check (using odd parity)
    - **During Reads** the 9th bit is called **Transition Bit or T-Bit**, and it tells the Controller whether the Target has more data to send: if the T-Bit is 1, then the Target has more data to send; if 0, then there's no more data to send.

      For Reads, the Target switches the SDA line to Hi-Z on the raising edge of the SCL signal. This allows the Controller to either continue or stop the transfer by issuing a STOP (i.e., holding and then raising SDA) or a Repeated START.

      For complete details regarding T-Bit operation, see I3C Specification ***Section 5.1.2.3.3*** and ***Section 5.1.2.3.4 [MIPI06]***.

5.1.2 Bus Transfers

In I3C a Controller initiates Bus Transfers, except for a few special cases such an In-Band Interrupt or Hot-Join. Controller-initiated Bus Transfers always start with the I3C Reserved Address, whose value 7'h7E is ignored by I^2C Targets. This makes it possible to detect whether there are any I3C Targets active on the Bus, and if so, to then switch to Push-Pull mode for a more efficient transfer.

I3C extends I^2C's basic Read and Write concepts by adding commands, called Common Command Codes (CCC) which may or may not have an associated Data Sequence. Immediately following the Reserved address ACK, a CCC is sent, possibly followed by Data Sequence.

Figure 1's first three illustrations (A, B, and C) show the three currently defined types of CCC Bus transfers:

- **A: Broadcast CCC Writes** perform a Write to all Targets that are active on the bus.
- **B: Directed CCC Writes** perform a Write to a single addressed Target.
- **C: Directed CCC Reads** perform a Read from a single addressed Target.

Note:

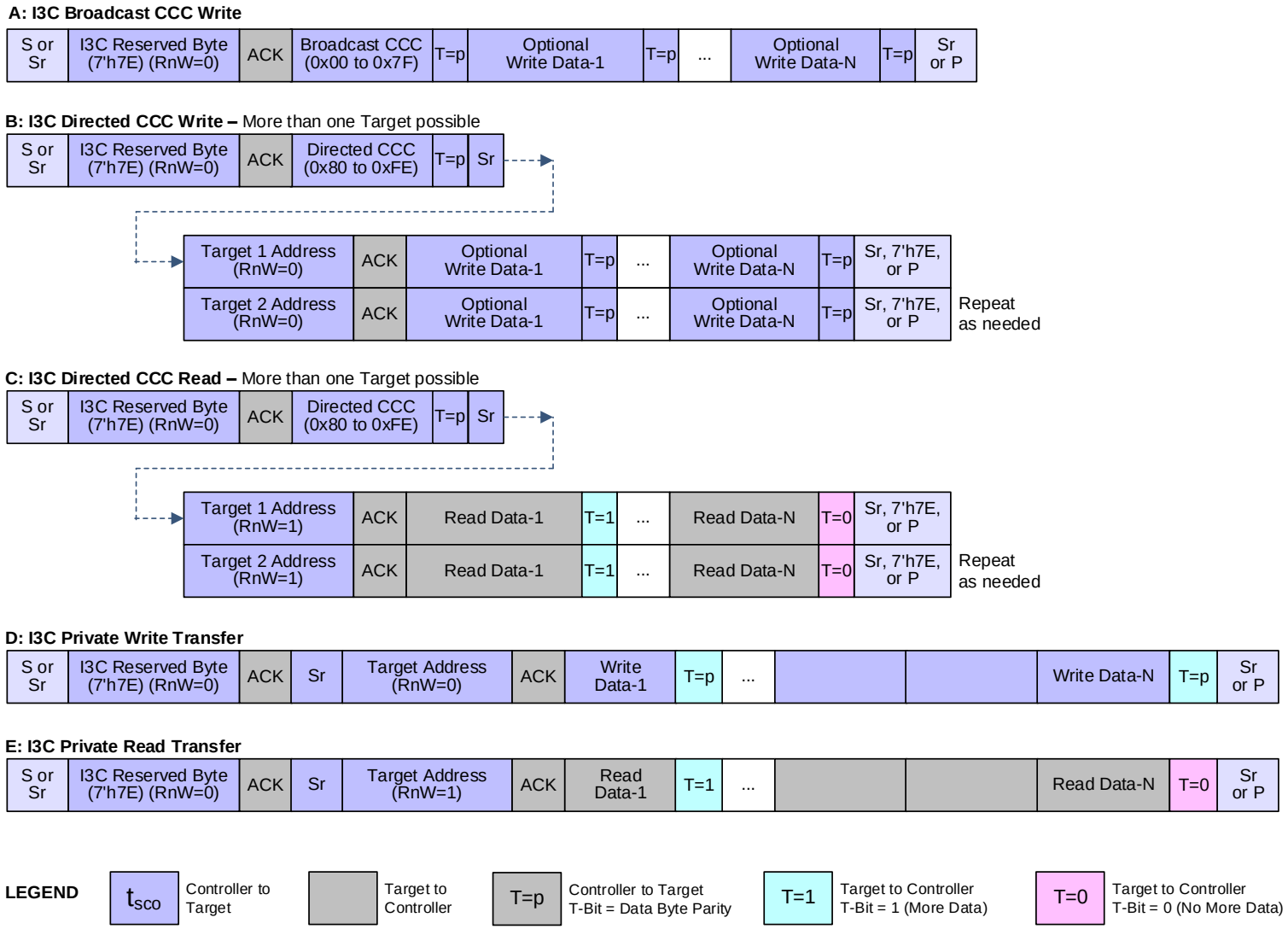
A rapid succession of Reads or Writes addressing different Targets can be executed by using Directed CCCs separated only by Repeated STARTs.

Sending a Directed CCC is not the only option to exchange data. I3C can also perform unstructured Write and Read transfers, called Private Transfers, where the data content is application specific.

Private Transfers take place after the Reserved byte 7'h7E with RnW=0, followed by a Repeated START. The Target Address is sent with a RnW bit properly set (see ***Figure 1*** illustrations D and E):

- **D: Private Write** transfer if RnW=0, followed by Data Sequence sent by the Controller, with T=odd parity.
- **E: Private Read** transfer if RnW=1, followed by Data Sequence sent by the Target, until T=0 (no more data).

As the above makes clear, I3C Bus transfers are more complex than those in I^2C.



209

Figure 1 I3C Transfer Types

5.1.3 SDA Line Discipline

I3C Controller and Target Devices must be able to dynamically switch the SDA line between Open Drain, Hi-Z, and Push-Pull configurations. During Bus idle, the SDA line is in Hi-Z to allow targets to initiate In-Band Interrupts, Hot-Join events, or Secondary Controller Requests. Following the START condition, during the initial 7-bit address (for the I3C Reserved Address 7'h7E, the IBI reserved address, or other address values) the SDA must be kept in Open Drain to allow address arbitration to occur. After the arbitration is concluded, SDA can be switched into Push-Pull, driven by either the Controller or the Target, depending on the Bus transfer stage.

Additionally, there are a few conditions where SDA must be kept in Open Drain or Hi-Z mode:

- During Dynamic Address Assignment (DAA), the 48-bit unique ID is issued in Open Drain mode to allow arbitration. DAA will occur at Bus initialization, or during a Hot-Join Request.
- During ACK bit time, to detect whether any Targets (i.e., at least one Target) is issuing an ACK.
- In a Read, during T-Bit time after the SCL raising edge, to allow the Controller to either continue or abort the transfer.
- During a Secondary Controller Request to become the Active Controller, to allow address arbitration in case of requests collision.

5.1.4 High Data Rates (HDR Modes)

In addition to Standard Data Rate (SDR) Mode, I3C also supports data transfer at higher speeds through the use of more sophisticated line coding, and/or multiple (x2 or x4) SDA lanes.

There are currently four supported HDR Modes.

Two HDR Modes are available in both the full I3C Specification and I3C Basic:

- **HDR-DDR Mode:** Double Data Rate, where data bits are clocked on every SCL edge transition (i.e., both the rising edge and the falling edge), effectively doubling SDR Mode data rate.
- **HDR-BT Mode:** Bulk Transport, an SDR-like mode that is block oriented, using the 9th bit for data instead of the usual Parity or T-Bit function (data integrity is checked with CRC instead). This results in a 20% performance gain over the base SDR Mode.

The two Ternary Modes are only available in the full I3C Specification, not in I3C Basic:

- **HDR-TSP Mode:** Ternary Symbol, Pure Bus, where SDA and SCL lose their ordinary functions and data is instead sent over both wires in the form of ternary symbols, achieving x3 speed gain. HDR-TSP Mode is not compatible with Legacy I^{2}C Devices, so it requires a 'pure' I3C Bus (only I3C Devices, no Legacy I^{2}C Devices).
- **HDR-TSL Mode:** Ternary Symbol, Legacy Bus. Similar to HDR-TSP Mode, but allows Legacy I^{2}C Devices to be present on the Bus. This comes at the cost of a slight performance reduction compared to HDR-TSP Mode: x2.5 speed gain, instead of 3x.

5.2 I3C Device Characteristics

Devices on an I3C Bus have different Roles, each bearing distinct responsibilities.

The defined Roles are:

- Primary Controller
- Secondary Controller
- Target
- SDR-Only Primary Controller
- SDR-Only Secondary Controller
- SDR Only Target

An active I3C Device playing a given Role in a given I3C Bus instantiation will fulfill all responsibilities for that Role, as detailed in the I3C Specification at ***Section 4.2*** (***Table 1 Roles for I3C Compatible Devices***) and ***Section 5.1.1.1*** (***Table 2 Devices Roles vs Responsibilities***) ***[MIPI06]***.

A given I3C Device may perform different Roles at different times. The most notable cases of this are (A) a Target that gains a Secondary Controller Role to accomplish a specific task, and then relinquishes the Role after completing the task, and (B) when an I3C Target Device connects after initialization (i.e., the Hot-Join Request), which may also change the electrical characteristics of the I3C Bus.

As a result, the configuration of a given I3C Bus can vary over time: it will depend upon the Roles and the states of the I3C Devices that are active on that I3C Bus at a given time. System Designers should anticipate these dynamic changes in the configuration of the I3C Bus, and account for them in their designs.

Device Roles are grouped into two sets, based on whether the Device supports HDR Mode. As the names imply, the SDR-Only Devices cover the Roles of I3C Devices that do not support any of the HDR Modes. By contrast, the Devices in the Roles of Primary Controller, Secondary Controller, and Target may choose which of the optional HDR Modes to support.

Note:

I3C Devices that are not HDR capable (or that only support some HDR Modes) are required (A) to be tolerant of the HDR Modes that they do not support, and (B)to recognize the HDR Exit Pattern.

SDR-Only Primary Controller Devices and SDR-Only Secondary Controller Devices are still required to support and use the HDR Exit Pattern, for certain error recovery situations. For details, see the I3C Specification at ***Section 5.1.10 [MIPI06]****.*

For the remainder of this Application Note, the differences between a Role and its SDR-Only counterpart will not generally be relevant; as such, the Roles of Primary Controller, Secondary Controller and Target are used in most situations where HDR Mode support is not relevant to a particular section.

5.2.1 Devices Roles and Responsibilities

5.2.1.1 Primary Controller Role

In an I3C Bus there can be only one Primary Controller Device. The Device designated as the Primary Controller keeps its Role at all times, even when it is not the Active Controller of the I3C Bus.

Note:

If the Primary Controller supports only SDR data transport, then it is referred as an SDR-Only Primary Controller. This Application Note will generically refer to the term Primary Controller, unless the "SDR-Only" qualifier is needed.

The Role of Primary Controller is the most complex in the I3C Bus. The Primary Controller has the authority for the initial configuration of the Bus and all Devices, including any Legacy I^2C Devices. It must be capable of handling all Bus Configuration procedures.

The required capabilities of the Primary Controller include:

- Assign Dynamic Addresses using the ENTDAA, SETDASA, and SETAASA CCCs (per the I3C Specification at ***Section 5.1.4 [MIPI06]***) to:
  - All I3C Targets, as part of Bus Initialization and any subsequent Bus Configuration, and to itself
  - Any Hot-Joining Devices that might not be present during Bus Initialization
- Maintain a memory map of assigned Dynamic Addresses for all I3C Targets, including their Bus Characteristics and configuration (i.e., BCR and DCR)
- Manage the SDA and SCL Bus Lines while serving as the Active Controller, including:
  - In SDR, HDR-DDR and HDR-BT Modes, driving the SCL line for data clocking
  - During the I3C Address Header, manage Address Arbitration per ***Section 5.1.2.2*** of the I3C Specification ***[MIPI06]***, especially when starting new transfers with the 7'h7E Broadcast Address
  - Generate the SDA ACK during a Hot-Join Request per ***Section 5.1.5*** of the I3C Specification ***[MIPI06]***, and during an In-Band Interrupt Request per ***Section 5.1.6***, as and when the Primary Controller is enabled to support such interrupts
  - HDR Mode management
  - Support data reception for read transfers in the optional HDR-TSP and/or HDR-TSL Modes, i.e., when passing control of both SDA and SCL to a particular I3C Target)
  - Generate the I3C Target Reset Pattern, per ***Section 5.1.11*** of the I3C Specification ***[MIPI06]***

If the I3C Bus also contains one or more Secondary Controller Devices, then the Primary Controller is required to have additional capabilities:

- Respond to Controller Role Requests from a Secondary Controller
- Pass the Controller Role to a chosen Secondary Controller, per ***Section 5.1.7.2*** of the I3C Specification ***[MIPI06]***:
  - Prepare the Bus for Handoff
  - Use the defined Controller-to-Controller Handoff Procedure to transfer the Controller Role
- After passing the Controller Role, remain in standby mode (the behavior is similar to a Secondary Controller) while the other Controller-capable device holds the Active Controller Role, until it is time to pass the Controller Role back again
- Transfer the assigned Dynamic Address memory map to any Secondary Controller Devices that might be on the Bus. This includes Group Address assignment for any I3C Targets that are members of Groups.

5.2.1.2 Secondary Controller and SDR-Only Secondary Controller Roles

In I3C, a Target can also hold the Role of Secondary Controller. For that, the Device should be capable of requesting and/or acquiring the Controller Role, i.e., of becoming the Active Controller. The Controller Role can only be passed after the Primary Controller has performed Bus Initialization.

The I3C Specification does not place any time limitation on when a Secondary Controller can become the Active Controller of the Bus, however, it is expected that for most applications, a Secondary Controller will remain Active only for the amount of time needed to perform a specific task. Once the task is complete, the Secondary Controller will relinquish the Active role, usually back to the Primary Controller, or potentially to another Controller-capable Device on the I3C Bus (the actual behavior will depend on the particular application).

Controller-capable Devices use the Controller-to-Controller Handoff Procedure that is managed by the current Active Controller, as described in ***Section 5.1.7.2*** of the I3C Specification ***[MIPI06]***. Note that this Controller-to-Controller Handoff Procedure is the only means by which an Active Controller can hand the Controller Role off to another Controller-capable Device (e.g., a Secondary Controller).

Note:

The Controller-to-Controller Handoff Procedure always has the same flow, no matter whether the Active Controller is the Primary Controller at Bus Initialization time vs. a Secondary Controller.

It is important to understand that I3C Controller-capable Devices do not directly contend with one another for the Controller Role. Instead, the Controller-to-Controller Handoff Procedure is fully managed by the Active Controller, using the GETACCCR CCC per I3C Specification ***Section 5.1.9.3.16 [MIPI06]****.*

The responsibilities of an Active Secondary Controller are similar to those of the Primary Controller:

- Manage the SDA and SCL Bus Lines while serving as the Active Controller
- Optionally Assign Dynamic Addresses to a Hot-Joining Device, using the ENTDAA CCC
- Receive and maintain a memory map of assigned Dynamic Addresses for some or all I3C Targets, including their characteristics and configuration (i.e., BCR and DCR)
  - If the assigned Dynamic Address memory map has changed while this Device was the Active Controller, then transfer the updated memory map to the Primary Controller

A Secondary Controller is also required to be capable of interacting with the Primary Controller and any other Controller-capable Devices:

- If the assigned Dynamic Address memory map has changed while this Device was the Active Controller, then transfer the updated memory map to any other Secondary Controller Devices and the Primary Controller. This includes changes to Group Addresses if any of them have changed.
- Respond to any Controller Role Requests that are received from Controller-capable Devices on the Bus (including the Primary Controller)
- Pass the Controller Role to a chosen Controller-capable Device, per the I3C Specification at ***Section 5.1.7.2 [MIPI06]***:
  - Prepare the Bus for Handoff
  - Use the defined Controller-to-Controller Handoff Procedure to transfer the Controller Role
  - After passing the Controller Role, remain in standby mode (i.e., act only as a Secondary Controller) while the other Controller-capable Device holds the Active Controller role, until such time as the Controller Role is passed back again.

5.2.1.3 Target and SDR-Only Target Role

Since I3C transactions are initiated by the Active Controller, the Role of I3C Target Devices is for the most part a passive one.

However, there are a few instances in which a Target plays an active role, generating signals to initiate specific actions:

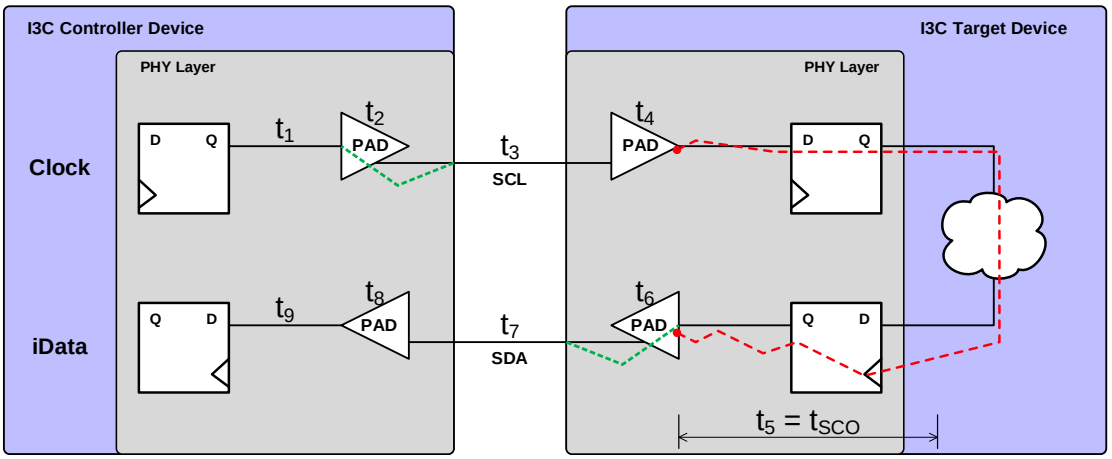
- To initiate an In-Band Interrupt request, a Target can pull the SDA line low (see the I3C Specification at ***Section 5.1.6 [MIPI06]***).
- To initiate a Hot-Join Request, a Target can pull the SDA line low (see the I3C Specification at ***Section 5.1.5 [MIPI06]***).
- A Target also manages the T-Bit:
  - For Writes, the T-Bit indicates parity from the Active Controller
  - For Reads, the Target uses the T-Bit to indicate whether additional data bytes are available: if the Active Controller drives the T-Bit to 0, then the Target will end the Read transfer.

If a Target is HDR-capable, then it is also required to:

- Support the appropriate ENTHDRx CCC to enter the HDR Mode, per the I3C Specification at ***Section 5.1.9.3.9 [MIPI06]***
- Recognize its assigned Address in the appropriate manner (i.e., a Command Word or Header Block, specific to the HDR Mode) and respond appropriately when the Address matches (i.e., providing ACK or NACK to the command)
- Support reception and generation of the different HDR signaling modes, for Write and Read commands initiated by the Active Controller
- Detect errors in transmission
- Recognize the HDR Exit Pattern

5.2.2 Clock-to-Data Turnaround Time (t_{SCO})

This section discusses the impact of Clock-to-Data turnaround (t_{SCO}) on a single-ended, single-clock Bus such as I3C when operating the interface over long traces (> 0.5 m). ***Figure 2*** illustrates this scenario. Ideally, there should be no timing skew between t_3 and t_7. To limit this timing skew, the SCL and SDA paths should be designed to be as similar to each other as possible.



t_1: Time from Clock Flop Q to Pad
t_2: Time through output pad (PFET/NFET)
(tested over 90 Ω , 50 pF line C)
t_3: Time over wires pad-pad: Controller drive + Line Cap + Tpath
t_4: Time through input pad of Target
t_5: Time inside Target, from SCL input to SDA out (t_{SCO})
(to gate drive of SDA)
t_6: Time through output pad of Target
t_7: Time over wires pad-pad: Target drive + Line Cap + Tpath
t_8: Time through iData input pad (Schmitt input)
t_9: Time from iData pad to D input for serializer

LEGEND
Internal delay (excluding pads)
PAD delays on a standard Bus model

Figure 2 Components of Clock-to-Data Turnaround Delay (t_{SCO})

Note:

__Figure 2__ also appears in __Section 5.1.9.3.18__ of the I3C Specification __[MIPI06]__ where the GETMXDS CCC is defined. GETMXDS allows an I3C Target to return its expected t_5 time.

5.2.3 Pad Capacitance

The pad capacitance adds to the capacitance of the Bus wires, and must be considered when computing the effective Bus frequency or other Bus parameters. A Device's pad capacitance also contributes directly to the skew of the signals, to the Device internal delay, and to the t_{SCO}. For this reason, t_{SCO} was characterized on a standard 50 pF, 90 Ω internal driver resistance.

5.2.4 Pad Drive Strength

The pad drive strength is another factor contributing to signal skew, and to the Device's ability to drive the pad within the required rise and fall time. The minimum recommended Bus drive strength is 4 mA drive. Greater drive strength can be implemented, as long as the Bus reflections and the power requirements of the system design are not affected. Most Bus timing parameters in the I3C Specification were determined using 4 mA drive strength as the measurement reference. When choosing greater drive strength, care must be taken to avoid overshoot.

5.2.5 BCR Use

Each I3C Device that is connected to the I3C Bus has an associated read-only Bus Characteristics Register (BCR). The BCR fields describe the Device's Role and capabilities that Primary and Secondary Controllers must consider when operating a Device on the Bus. The BCR is described in the I3C Specification at ***Section 5.1.1.2.1***, in ***Table 5 Bus Characteristics Register [MIPI06]***, reproduced below as ***Figure 3*** (table notes omitted).

Bit	Name	Description	Notes
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	1
BCR[6]	**Device Role[0]**		
BCR[5]	**Advanced Capabilities**	**1:** Supports optional advanced capabilities. Use GETCAPS CCC (***Section 5.1.9.3.19***) to determine which ones. **0:** Does not support optional advanced capabilities	2
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)	3
BCR[3]	**Offline Capable**	**0:** Device will always respond to I3C Bus commands **1:** Device will not always respond to I3C Bus commands	4
BCR[2]	**IBI Payload**	**0:** No data bytes follow the accepted IBI **1:** One data byte (MDB) shall follow the accepted IBI, and additional data bytes may follow; see also the Set/Get Maximum Read Length CCC (***Section 5.1.9.3.6***). Data byte continuation is indicated by T-Bit per ***Section 5.1.2.3.4***. See also ***Section 5.1.8*** on 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	5

Figure 3 Bus Characteristics Register (BCR)

- If **BCR bit [0]** is set, then one or more of the following limitations apply:
  - The Device is not capable of operating at the 12.5 MHz maximum operating frequency
  - The time between the end of the SCL falling edge and start of the SDA output from the Target is greater than 12 ns
  - Overall internal Device delays, including Pads (see ***Figure 2***), is greater than 12 ns
  - The Device requires more time to prepare the requested data. As a result, the Controller will have to send a GETMXDS command to communicate a lower speed that is acceptable by all Devices to determine the particular limitation(s), and then take the appropriate measure(s) necessary to accommodate the Device.

- **BCR bits[2:1]** are related to In-Band Interrupt. If BCR bit [1] is set, then the Target is capable of issuing IBI Requests. If BCR bit [2] is also set, then during the IBI event the Controller will continue to generate SCL clock pulses to allow the Target to return the IBI Mandatory Data Byte, followed by additional data, until the data transfer is terminated (either terminated by the Target itself via the T-bit, or terminated by the Active Controller).
- If **BCR bit [3]**, the Offline Capable bit, is set, then the Device is capable of going offline (i.e., of entering a state in which it will not respond to commands, but still retains its Dynamic Address).
- If **BCR bit [4]**, the Virtual Target Support bit, is set, then the Device is capable of presenting multiple Virtual Targets, or capable of exposing other downstream Devices (i.e., from another I3C Bus segment, or from any other Bus type)

  Note:

  *The meaning of BCR bit [4] has changed from earlier versions of the I3C Specification. For additional requirements for Virtual Target support, see the I3C Specification at **Section 5.1.2.1.2 [MIPI06]**, and the **Application Note for Virtual Devices & Virtual Targets [MIPI08]**.*
- If **BCR bit [5]**, the Advanced Capabilities bit, is set, then the Device supports optional advanced capabilities, such as I3C spec supported version, Device-to-Device Transfer, Group Address, Multi-Lane, and others. The Active Controller can discover which of these optional features are supported by using the GETCAPS CCC per the I3C Specification at ***Section 5.1.9.3.19 [MIPI06]***. (The complete set of optional features is listed in that section.).

  Note:

  The meaning of BCR bit [5] has changed from earlier versions of the I3C Specification. Previously, this bit only indicated whether the Device supported any optional HDR Modes.

  Per earlier versions of the I3C Specification, certain SDR-only Devices might return a 1'b0 value in this bit. However, the I3C Specification now requires all conforming Targets and Secondary Controller to return a 1'b1 value, as the GETCAPS CCC now requires such Devices to report other advanced capabilities as well as the I3C Version number that is supported. In practice, all new I3C Target and I3C Secondary Controller implementations must return a 1'b1 value, and must support the GETCAPS CCC.
- **BCR bits[7:6]**, the Device Role bits, indicate whether the Device is simply an I3C Target, or whether it also has Controller Role capabilities.

5.3 Dynamic Address Assignment (DAA)

I3C device addressing is different from I²C, where devices have hardwired addresses that are fixed, or set by pin strapping. In I3C, a device address is dynamically allocated at Bus initialization time when the Primary Controller issues the ENTDAA CCC. Dynamic Addresses (DAs) can also be assigned subsequently (i.e., after the Bus has been initialized) for some specific cases, such as for a Hot-Joining device. A I3C component becomes fully functional only after it has acquired its DA. The pool of Dynamic Addresses is 7-bit wide; more insights on this topic are given below in ***Section 5.3.3*** of this Application Note, ***Address Assignment Guidance***.

From the system design perspective, I3C's Dynamic Addressing scheme brings numerous advantages:

- SW drivers and application management can be simplified by associating a given Dynamic Address with a logical function (e.g., a magnetometer), rather than with a specific hardware component.
- The relatively small address space (7 bits) is not a limitation, as it is tied to Bus functions, not to different vendor HW models.
- The short, and therefore fast, 7-bit address is an efficient way to address a Target on a I3C Bus.
- The DA value determines the priority ranking in several I3C Bus transactions. For In-Band Interrupts (IBI), a lower DA value has higher priority, hence the DAA strategy regulates the servicing order for concurrent IBI requests.

The initial assignment of Dynamic Addresses is the duty of the Primary Controller: as soon as the I3C Bus is powered up, the Primary Controller assigns a unique Dynamic Address to each of the connected Targets. The Primary Controller has also a unique position among other possible Controller-capable Devices that might be connected to the Bus: it is constantly informed of the number of connected components and their characteristics.

5.3.1 Dynamic Address Assignment Procedure

The Primary Controller starts the DAA procedure once all of the Devices on the I3C Bus are ready.

The DAA procedure is really a two-step process. First, every static address is mapped to a DA as explained below. Then the Primary Controller performs the actual DAA operation by issuing the ENTDAA CCC with the 7'h7E reserved Broadcast byte, followed by a Repeated START (see ***Figure 4***).

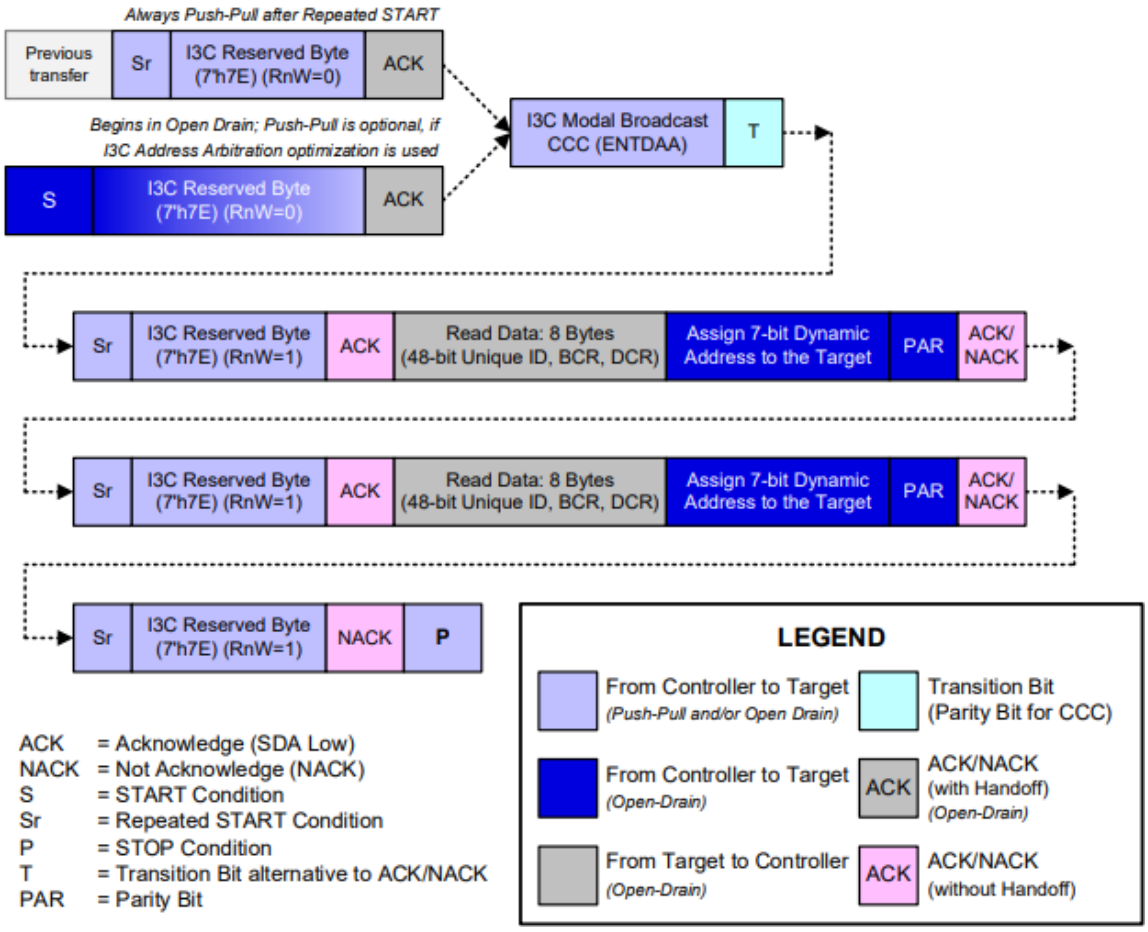


Figure 4 Dynamic Address Assignment Transaction

All I3C Devices that have not yet received their Dynamic Addresses will respond to the ENTDAA CCC sequence by ACKing the reserved byte, and then sending their 48-bit Provisioned ID which is both unique and arbitrable. Although multiple I3C Devices will ACK the CCC, the Primary Controller will only assign a 7-bit DA value to the one with the lowest-value Provisioned ID (this is "winning the arbitration"). The winning Device will then not ACK the next ENTDAA CCC, because then it will have a DA. This assignment cycle is repeated until the Primary Controller receives a NACK to the reserved byte, because that indicates that no further Targets are still requesting a DA. The Dynamic Address Assignment procedure ends with a STOP: a robust signaling condition that all Devices connected to the I3C Bus can easily identify. A higher-level I3C Bus Management layer is responsible for managing the whole operation.

Optionally, the Primary Controller can end the Dynamic Address Assignment procedure at any time. To complete the DAA procedure, any Devices that haven't yet received an assigned I3C Dynamic Address will have to participate in a further Dynamic Address Assignment procedure which the Primary Controller will have to schedule at a later time.

All the details of the Dynamic Address Assignment (Dynamic Address Assignment) procedure are described in the I3C Specification at ***Section 5.1.4.2 Bus Initialization Sequence with Dynamic Address Assignment [MIPI06]***.

After the initial Bus configuration is completed, the Active Controller can still change a Target's Dynamic Address using the SETNEWDA CCC per ***Section 5.1.9.3.7*** of the I3C Specification ***[MIPI06]***, if the Target supports this CCC. In addition, the Active Controller will assign a Dynamic Address to each Target that is newly attached to the Bus via a Hot-Join Request.

5.3.2 Dynamic Address Consistency

Secondary Controllers need to be informed of the correspondence between the assigned Dynamic Addresses and their associated Targets. This information can be communicated in two ways:

- **Passively:** The Secondary Controller(s) monitor the Dynamic Address Assignment procedure, by snooping the data for the CCC Bus transactions for the RSTDAA, ENTDAA, and/or SETNEWDA CCCs, or
- **Actively:** The Primary Controller issues the Broadcast DEFTGTS CCC (***Define List of Targets)*** per ***Section 5.1.9.3.7*** of the I3C Specification ***[MIPI06]***.

The passive method has several limitations. It only works if the Secondary Controller is powered and active during Bus Initialization. If there are any Targets that use their I^2C Static Address as their DA (this is configured via the SETAASA CCC), then the Secondary Controller will not see the I^2C address (SETAASA has not optional data), and they will remain unknown. Also, if the Secondary Controller goes into an idle or powered-down mode, then it might miss later DAA procedures with the ENTDAA CCC.

The active method, though more complex, is more reliable than the passive method.

5.3.3 Address Assignment Guidance

Proper selection and allocation of the Dynamic Addresses is crucial for the functioning of the I3C Bus. Although there is a maximum of 128 possible Dynamic Addresses (because the address is 7 bits wide), in practice the typical number of physical Devices present on a I3C Bus is considerably less than that, partly because the maximum capacitance limit is typically reached well before 128 Devices can be placed on the Bus (i.e., the combined capacitances of the I/O pins and the Bus wiring is 50 pF for a typical system).

Note that it is possible for the number of assigned, unique Dynamic Addresses to be greater than the number of Devices physically present on the Bus. This can happen because besides the physical Device addresses, the DA count will also include one additional DA for each Virtual Target (per I3C Specification ***Section 5.1.2.1.2 [MIPI06]***) and one additional DA for each Group Address (***Section 5.1.2.1.3***) that those Devices implement.

5.3.3.1 Effective Address Space

Though the Dynamic Address is 7 bits wide, the number of usable Dynamic Addresses is less than 128 because several values are either reserved in the I3C protocol, or cannot be used under certain conditions.

The I3C Specification details the unavailable addresses at ***Section 5.1.2.2.5 I3C Target Address Restrictions***, but in summary:

- The I3C Broadcast Address 7'h7E is not available as a Dynamic Address because in the I3C protocol the 7'h7E value serves as the preamble for virtually all Bus transactions
- To make Bus operation more robust against single-bit errors, all addresses that are a 1-bit hamming distance away from the 7'h7E Broadcast Address (i.e., to avoid 1-bit errors) are excluded: 7'h3E, 7'h5E, 7'h6E, 7'h76, 7'h7A, 7'h7C, and 7'h7F
- I3C reserves 7'h02 for the Hot-Join address
- I3C reserves 7'h00 (its use is forbidden in I3C)
- I3C reserves 7'h01 for SETDASA CCC point-to-point communications
- If any I^2C Devices are present on the Bus, then the I^2C reserved address 7'h03 is excluded
- If certain types of Legacy I^2C Devices are present on the Bus, then the following I^2C-related addresses are also excluded:
  - I^2C Devices supporting High-Speed Mode: 7'h04, 7'h05, 7'h06, 7'h07
  - I^2C Devices supporting Extended Address Mode or having an Extended Address: 7'h78, 7'h79, 7'h7B
  - I^2C Devices supporting Device ID Mode: 7'h78, 7'h79, 7'h7B

5.3.3.2 Address Allocation Good Practice

An optional, but recommended, practice to speed up the Dynamic Address Assignment procedure is to first assign Dynamic Addresses to all Devices with a known Static I^2C Address via the CCC ***Set Dynamic Address from Static Address (SETDASA)*** (***Section 5.1.9.3.10*** of the I3C Specification).

Once the Static addresses are set, the actual selection and assignment of Dynamic Addresses takes place. For the following reasons, it is a good practice to leave spaces (i.e., unused addresses) in between the assigned addresses:

- To allow for further changes in the priority ranking for In-Band Interrupt (IBI) processing
- To reserve pre-interleaved locations for Devices that will Hot-Join the Bus, thus facilitating their IBI priority ranking order.

As a further option, the Dynamic Addresses can be selected and allocated in the manner described at ***Section 5.1.2.2.2*** of the I3C Specification, ***I3C Address Arbitration Optimization***. The method described there can improve effective data rates by minimizing the amount of time spent in address arbitration operations.

5.3.3.3 Dynamic Address Collision

The Dynamic Address Arbitration procedure relies on the I3C Devices on the Bus having a mutually unique 48-bit Provisioned ID. However, since this value is generated using random data, it is possible for two (or more) Targets to independently generate the same Provisioned ID value. Though the probability of this actually occurring is quite small, if it does occur then the Controller will assign the same Dynamic Address to those two (or more) Devices. This is known as an address collision. An address collision might also potentially result if a signal integrity problem causes misreading.

Detecting whether an address collision has occurred is simple: the total number of I3C Dynamic Addresses actually assigned will be less than the number of Devices known to require a Dynamic Address.

To recover from an address collision, the Primary Controller may issue the CCC ***Reset Dynamic Address Assignment (RSTDAA)*** (I3C Specification ***Section 5.1.9.3.3***), which causes every I3C Device on the Bus to reset (i.e., to clear) the Dynamic Address it was just assigned. The Controller then re-initiates a new Dynamic Address Assignment procedure, in the hope that the Provisioned IDs will be mutually unique this time. If a collision reoccurs more than a given number of times (three is recommended), then the Primary Controller is required to inform the Application Layer that the I3C Bus is not functional (see the I3C Specification at ***Section 5.1.4.3***).

5.3.3.4 Dynamic Address Modification

After the I3C Bus has been successfully configured, the Active Controller can dynamically change assigned Dynamic Addresses as desired for optimal operation of the running application, using the CCC ***Set New Dynamic Address (SETNEWDA)*** (I3C Specification ***Section 5.1.9.3.11***). If any Secondary Controllers are connected to the I3C Bus, then the Active Controller must inform them of any Dynamic Address changes by using the ***Define List of Targets (DEFTGTS)*** CCC (I3C Specification ***Section 5.1.9.3.7***).

Note:

> *Certain I3C Targets might not support the SETNEWDA CCC, and will only support an initially Dynamic Address that cannot be changed after it is assigned (i.e., during Bus Initialization). If such I3C Targets receive their Dynamic Address from their I^2C Static Address (i.e., using the SETDASA CCC or the SETAASA CCC), then the Dynamic Address is effectively immutable and the Active Controller must manage the assignment of Dynamic Addresses accordingly: for IBI prioritization, the Active Controller must assign Dynamic Addresses for other I3C Targets to work around such immutable Dynamic Addresses for these I3C Targets.*

In general, the Active Controller should assign a Dynamic Address to Hot-Joining Targets. However, not all Secondary Controllers are capable of assigning Dynamic Addresses. If a Target attempts to Hot-Join while such a Secondary Controller is the Active Controller, then that Secondary Controller is required to pass the

Active Controller Role back to the Primary Controller (or to some other Controller known *a priori* to be sufficiently capable). That new Active Controller will then furnish the Hot-Joining Target with the Dynamic Address it needs, and then inform any other Controller-capable Devices on the Bus (i.e., Secondary Controllers) of the new address assignment via the ***Define List of Targets (DEFTGTS)*** CCC (I3C Specification ***Section 5.1.9.3.7***).

As emphasized in the I3C Specification, the thing that distinguishes a Device participating on the Bus as an I3C Device from a Device that remains in its initial power-on state is whether it has an I3C Dynamic Address. This distinction is particularly important when the I3C-capable Device is also capable of acting as an I^2C Device on a Legacy I^2C Bus. Such Devices will have their 50 ns Spike Filters enabled and active on initial power-on, until they know they are on an I3C Bus. This Spike Filter must be taken into account by the Host layer (i.e., the higher layer that controls the Primary Controller, and through it the I3C Bus) in performing Bus Initialization and initiation of the Dynamic Address Assignment procedure (via the ENTDAA, SETDASA, or SETAASA CCC). Devices that power-up with the Spike Filter in effect won't know that they're on an I3C Bus (as opposed to a Legacy I^2C bus) until they receive an I3C Dynamic Address.

The Host layer and Controller must also take into consideration the fact that the I^2C-compatible Devices will re-enable their 50 ns Spike Filters after the RSTDAA CCC is sent. This means that the Active Controller must send a valid I3C Address Header with the 7'h7E Broadcast Address at a speed sufficiently slow to be received even with the Spike Filter enabled. Only then will such Devices recognize that they are on an I3C Bus, and as a result disable their Spike Filters (see ***Section 5.1.2.1.1*** of the I3C Specification ***[MIPI06]***).

Note:

*I3C Target Devices that use the standard Hot-Join method (see **Section 5.1.5** of the I3C Specification **[MIPI06]**) will not have the 50 ns Spike Filter, because the I^2C specification does not support Hot-Join functionality, and such Target Devices would assume by default that they are on an I3C Bus.*

5.4 Use of Legacy I²C Devices

By design, the I3C protocol can operate on an I3C Bus where Legacy I²C Devices are also connected. However, not all Legacy I²C Targets are fully supported. When using Legacy I²C Devices on an I3C Bus, System Designers must ensure that compatibility requirements are satisfied, and that the legacy devices do not significantly degrade I3C Bus performance.

The primary consideration in this regard is that most Legacy I²C Devices found in the market are designed to ignore SCL High pulses that are shorter than 50 ns. This feature is known as the Spike Filter. The presence of the Spike Filter allows many I3C Bus transactions to execute at higher speed.

Note:

*In version 7.0 of the I²C Specification **[NXP01]**, the SCL high pulse t_{HIGH} must be greater than 260 ns for Fast-mode+, greater than 60 ns for High-speed with 100 pF Bus load, or greater than 50 ns for Ultra Fast-mode.*

The I3C Specification defines several categories of Legacy I²C Devices and characterizes how an I3C Bus will perform if they are present (see ***Table 4 Legacy I²C-Only Target Categories and Characteristics*** in ***Section 5.1.1.1***, and ***Table 7 Legacy I²C Virtual Register (LVR)*** in ***Section 5.1.1.2.3 [MIPI06]***).

5.4.1 Considerations for Legacy I²C Devices

The I3C Specification classifies Legacy I²C Devices into three categories based on how they affect I3C Bus performance:

- **Index 0 Devices** have a Spike Filter that will cause them to ignore any SCL High pulse shorter than 50 ns.

  If Index 0 I²C Devices are present on the Bus, then the Active Controller can still communicate with I3C Devices on the Bus at up to the maximum SCL clock frequency (f_{SCL}), as long as the SCL High pulses for such transfers are shorter than 50 ns.

- **Index 1 Devices** will not have a Spike Filter *per se* but are tolerant of the maximum SCL clock frequency.

  If Index 1 I²C Devices are present on the Bus, then they will ignore any transfers to I3C Devices. The Active Controller can still communicate with I3C Devices on the Bus using up to the maximum SCL clock frequency (f_{SCL}), as long as the SCL High pulses for such transfers are shorter than 50 ns.

- **Index 2 Devices** will not have a Spike Filter *per se* and they are not tolerant of the maximum SCL clock frequency.

  Index 2 devices will significantly downgrade the Bus performance: If Index 2 I²C Devices are present on the Bus, then the Active Controller must restrict all transfer rates to the maximum SCL clock frequency that the Index 2 I²C Devices support. This limits the performance of the I3C Bus and can also prevent some HDR Mode transfers with minimum clock requirements.

For optimal performance of the I3C Bus, the System Designer should entirely avoid the use of Index 2 Devices. If the use of Index 2 devices cannot be avoided, then they should be separated from the I3C Devices by placing the Index 2 devices on a downstream Bus segment (i.e., using a Bridge Device) where they will not be able to impact transfer rates on the I3C Bus segment.

5.4.1.1 Detecting the Presence of the 50 ns Spike Filter

This section describes a possible method for detecting whether a given I²C Device has a 50 ns Spike Filter, or will properly tolerate (i.e., will ignore) higher-speed I3C transfers. (Other methods are also possible).

The test is done by sending a private read/write communication after sending the I3C Broadcast Address (see ***Figure 5***). The signaling marked in grey is handled as Open Drain transfers at an I²C-compliant SCL frequency, and the signaling marked in blue is handled as a higher-speed transfer where either the SCL High width (i.e., parameter t_{DIG_High}) is less than 50 ns, or I3C Mixed Bus timing is maintained.

START	7-bit Broadcast Address	RnW=0	ACK from DUT	
Repeated START	7-bit I²C Address	RnW=0	NACK from DUT	STOP

Figure 5 Spike Filter Detection Pattern

From the Legacy I²C Device's response to its I²C Static Address after the Repeated START, the Controller will know whether the device is capable of properly tolerating (i.e., ignoring) higher-speed transfers:

- **ACK:** A device that responds with ACK is not capable of ignoring higher-speed transfers, either because it has no 50 ns Spike Filter, or for some other reason: it successfully detected and processed the high-speed Bus transfer. Such a device would likely be Index 2.
- **NACK:** If the Controller receives NACK, then the device properly ignored the higher-speed transfer, either because it possesses a 50 ns Spike Filter (making it an Index 0 device), or for some other reason (making it an Index 1 device): it failed to detect and process the higher-speed Bus transfer.

Note:

Legacy I²C Devices will not respond to the I3C Broadcast Address (7'h7E / W), and the Bus might have a combination of Legacy I²C Devices and I3C Devices.

This also applies to I²C/I3C capable Devices which are supposed to respond to I²C commands directed to them using their I²C Static Address. If no Dynamic Address is assigned to these Devices and they don't know that they are on an I3C Bus (per ***Section 5.1.2.2.2*** of the I3C Specification ***[MIPI06]***), then their 50 ns Spike Filters will be engaged and they will behave as Legacy I²C Devices in all other respects. However, if such Devices detect a START followed by the I3C Broadcast Address (i.e., with their Spike Filter initially enabled), then such Devices will disable their Spike Filters and act as I3C Devices.

Note:

*A previous version of this Application Note indicated that such I²C/I3C capable Devices could determine whether to act as I3C Devices based on whether they had been assigned a Dynamic Address. However, more recent versions of the I3C and I3C Basic Specifications **[MIPI06]** have clarified the requirements. Per **Section 5.1.2.2.2**, the Controller is now required to transmit an SDR Frame with START followed by the I3C Broadcast Address (7'h7E / W) at slower speeds, in order to allow such I²C/I3C capable Devices to detect that they are actually on an I3C Bus, disengage their Spike Filters, and act as I3C Devices. As a result, the test procedure above would need to be applied **before** the Controller transmits the required START followed by Broadcast Address at slower speeds, so that such Devices can disengage their Spike Filters after the test procedure. Alternatively, the test procedure might not be necessary.*

5.4.2 I²C Clock Stretch is Not Allowed in I3C

In I²C the Controller drives the SCL line in Open Drain. This creates the possibility for I²C Target Devices to delay the I²C Bus when desired, by holding the SCL line Low. Delaying the I²C bus in this way is called "clock stretching".

In I3C, by contrast, the Active Controller generally drives the SCL line in Push-Pull. This brings several advantages and allows the I3C Bus to be optimized Bus for speed, efficiency, and loading. However, it also means that I3C Target Devices are not permitted to hold the SCL line Low. As a result, I²C-style clock stretching is not allowed in I3C.

5.4.3 Legacy Virtual Register (LVR) Use

Every Legacy I²C Device that can be connected to the I3C Bus has an associated read-only Legacy Virtual Register (LVR) describing the device's significant features. Since these are Legacy I²C Devices, it is understood that this I3C-specific register will not actually exist on the device as hardware. Instead, the LVR is expected to exist virtually, for example as part of the software driver for the device. When a Legacy I²C Device is present on an I3C Bus, fields in its LVR indicate what I²C Modes the device supports, and its maximum SCL clock frequency. The LVR fields are defined in the I3C Specification in ***Table 5 Bus Characteristics Register*** at ***Section 5.1.1.2.3 [MIPI06]***.

In particular, LVR Bits[7:5] define the device's Index value, which indicates whether such the device has a Spike Filter (can otherwise tolerate Bus transfers at up to the maximum I3C clock speed). The Host that directs the Primary Controller must have this knowledge before any Controller-capable Device attempts to drive I3C transfers. The reason is that if any Legacy I²C Devices have no Spike Filter (or for any other reason cannot tolerate Bus transfers at the maximum I3C clock speed), then the performance of all I3C transfers on the Bus will be impacted, and the Controller(s) will need to limit the use of SDR transfers at maximum clock speed. Additionally, such constraints might limit which HDR Modes and clock speeds can be used.

LVRs are typically transferred to the Primary Controller of the I3C Bus before the Bus is configured. As a result, the Primary Controller knows the content of the LVR for every I²C Device at Bus configuration, and this information can be transferred to any Secondary Controller(s) present on the I3C Bus by using the ***Define List of Targets (DEFTGTS)*** CCC (see ***Section 5.1.9.3.7*** in the I3C Specification ***[MIPI06]***).

5.5 I/O Characteristics

5.5.1 Pad Capacitance

As previously stated, the pad capacitance adds to the whole Bus capacitance and needs to be considered when computing the effective Bus frequency or other Bus parameters. The pad capacitance of a Device also contributes directly to the skew of the signals that the Device drives on the Bus, the Device internal delay, and the t_{SCO}. For this reason, it is important that Device vendors accurately specify the maximum pad capacitance such that System Designers can optimize I3C Bus performance for the total capacitance seen on the SDA and SCL lines.

5.5.2 SDA Drive Strength

As previously discussed, the pad drive strength is another factor that contributes to the signal skew and to a Device's ability to drive the pad within the required rise and fall time. Notably, I^2C (Fm/Fm+) and I3C can have different specified drive strengths, thus care should be taken to minimize any impact upon Bus performance due to reflections, overshoot, etc.

Most Bus timing parameters in the I3C Specification were developed using a 90 Ω output drive impedance as the measurement reference, and under those conditions the minimum recommended Bus drive strength is 4 mA.

Greater drive strength can be implemented, as long as the Bus reflections and system power requirements are not affected, and care is taken to avoid overshoot.

5.6 Bus Topologies

5.6.1 Bus Topology Types

5.6.1.1 Basic Bus Topologies

For purposes of illustration, this section of the Application Note defines and discusses three Bus topologies: Multi-Drop, Point-to-Point, and Star-on-Stick. If needed for a particular system design or use case, a given I3C Bus might use more than one of these Bus topologies together (see ***Section 5.6.1.2***). When calculating the distance between a Controller and its Target(s), a System Designer should consider all three Bus topologies.

- **Multi-Drop Bus Topology** (***Figure 6***). A single Controller is connected to two or more Targets A and B (etc.), all attached on the same branch.

  The distance of the medium between the Controller and Target A is **L1**, and the distance from Target A to Target B is **L2**; thus, Target B's distance **LT** from the Controller is **L1 + L2**.

  Figure 6 also shows each Target connected through a stub with length **St**, and after the stub another medium of length **L3**. As a result, target A's distance from the Controller is **L1 + St + L3**, and Target B's distance from the Controller is **L1 + L2 + St + L3**. Stub length plays an important role in signal integrity of the I3C Bus signals (SDA and SCL).

  The medium could be board trace only, or board and cables.

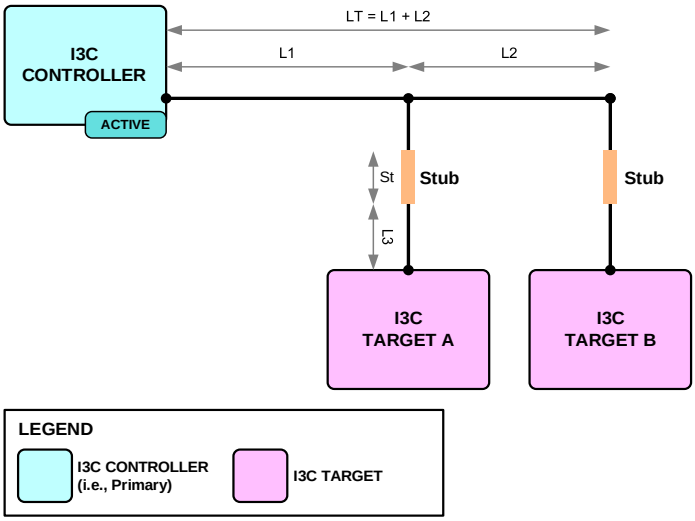


Figure 6 Multi-Drop Bus Topology

- **Point-to-Point Bus Topology** (***Figure 7***) has one Controller connected to one Target.

  For this topology, the two Devices are connected by a board trace of Length **LT**.

  The medium could be board and cable, board trace only, or any other medium.

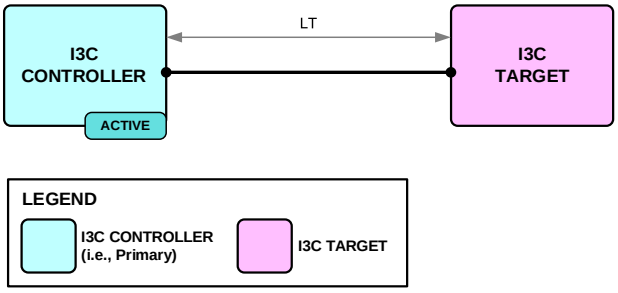


Figure 7 Point-to-Point Bus Topology

- **Star-on-Stick Bus Topologies** (***Figure 8***) are for Multi-drop Busses where all of the Targets are far apart from the Controller, but also close to each other (i.e., **L1** is much greater than **L2**).

  For this topology, the distance from the Controller to each Target is **L1 + St + L2**, and **L1** is much greater than **L2**.

  Although ***Figure 8*** shows equal distances for each of the Target Devices, in practice there could be per-Target distance variations. The System Designer must make sure that the distance for each Target is as equal as possible. In Star-on-Stick topologies, stubs also play an important role in I3C Bus signal integrity (SDA and SCL).

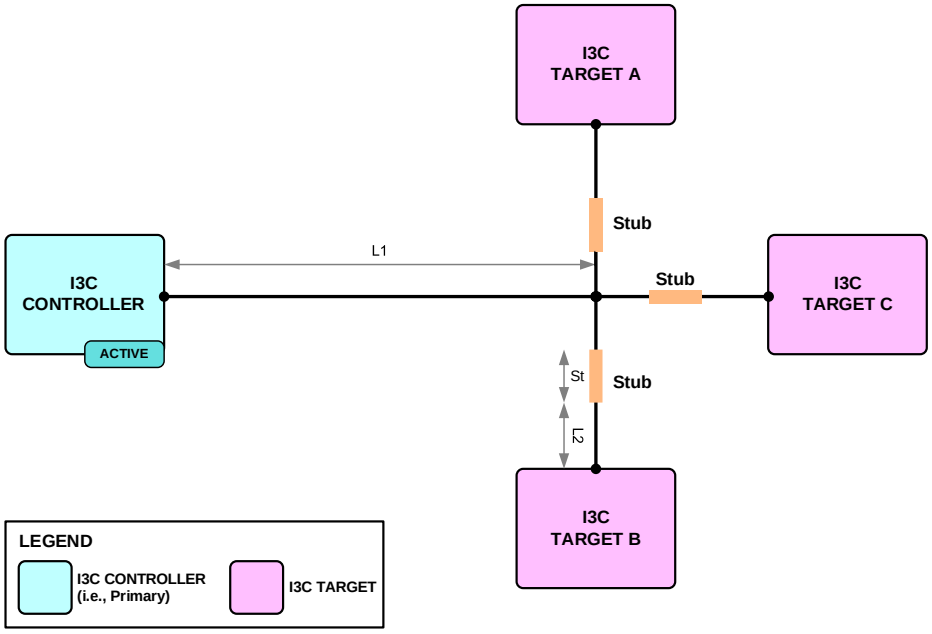


Figure 8 Star-on-Stick Bus Topology

5.6.1.2 Complex Bus Topologies

By combining the principles of these three basic Bus topologies, a System Designer could extend an I3C Bus to construct more complex Bus topologies, and optionally include more advanced I3C Devices. ***Figure 9*** shows an example I3C Bus that includes a simple I3C Target Device, a composite I3C Device that contains multiple I3C Targets (i.e., as multiple dies or wafers with internal traces or wires), and a more advanced I3C Device that presents multiple Virtual Targets using Shared Peripheral logic. (For more on Virtual Targets, see the separate ***MIPI Alliance Application Note for Virtual Devices and Virtual Targets [MIPI08]***).

- In this example, the per-Device distance from the Controller to each physical Device could be the same (i.e., based on the ideal Star-on-Stick Bus topology), whereas the length of the internal traces or wires inside the composite Device might vary. This will depend on the per-Target distance from the Device's external pins to the I3C Target's pads. In such a configuration, the theoretical distance from the Controller to each integrated Target would vary, but would be at least **L1 + St + L4** plus the actual per-Target trace or wire distance inside the integrated Device. If these per-Target trace/wire distances differ for each integrated Target, then the Bus topology becomes more complex.
- Alternately (although this is not a recommended configuration), the multi-Target integrated Device might present separate external I3C pins for each I3C Target.
- By contrast, an advanced I3C Device that uses Shared Peripheral logic to present multiple Virtual Targets has a simpler Target distance calculation (e.g., **L1 + St + L3**) since the Virtual Targets are not part of this calculation. For such a Device the Shared Peripheral logic would handle the I3C transfers, so the factor to consider would be the internal trace or wire distance between the pins and the Shared Peripheral's I/O pads. Device manufacturers should ensure that the I/O pads and Shared Peripheral logic are as close to the pins as possible.
- In all of the above cases, I3C Device manufacturers that make either multi-Target integrated Devices or composite Devices should publish the Device's internal trace or wire lengths, so that System Designers can have adequate information to calculate the per-Target length and understand the actual Bus topology.

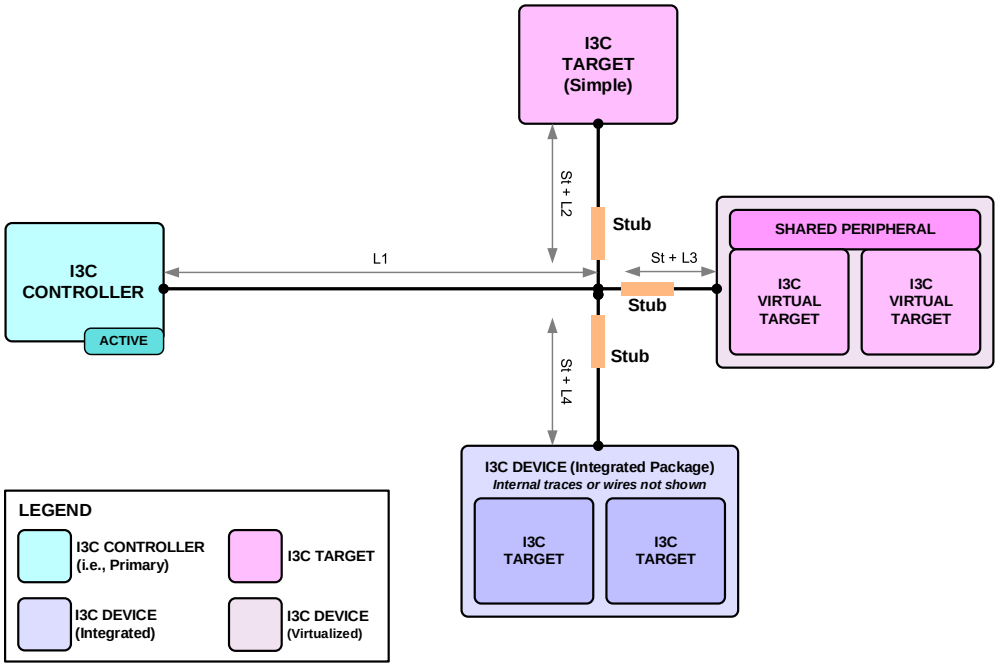


Figure 9 Complex Bus Topology

5.6.2 Trace/Medium

In the basic Bus topologies shown in ***Figure 6*** through ***Figure 8***, the medium connected to the Devices could be FR4 board trace, or board trace + cable, or trace + via + trace + cable, etc. The System Designer must make sure that the length of the trace or medium is chosen to meet the maximum Bus capacitance supported by the I3C Specification.

Example: A Point-to-Point Bus topology targeting 20 inches will result in 44 pF of board trace capacitance, where FR4 is 2.2 pF / inch.

Note:

The case of FR4 trace + Cable will result in different medium capacitance, and the calculation will not be direct as compared to single FR4 trace, because the discontinuity in the signal board trace also impacts the capacitance.

5.6.2.1 Length

The length of board trace that can be supported depends upon various factors, key among them being:

- **Bus Topologies:** Bus topologies impact the length supported for SDR/DDR mode and for HDR mode. Point-to-Point topologies will permit longer board traces or medium, assuming fixed capacitance from load.
- **Device Capacitance:** Device capacitance impacts Bus performance and maximum length for the targeted medium. The number of Devices present on the Bus will also increase the Bus capacitance, thus reducing medium length.
- **Reflections:** Signal integrity will affect the targeted medium length. The System Designer must be sure to avoid reflection or discontinuity, which will directly reduce targeted Bus length.
- **Medium:** Selection of a lossy medium will reduce targeted board trace length.

5.6.2.2 Material

The I3C Specification ***[MIPI06]*** does not define the physical conducting medium material: it can be PCB traces, cables, and/or connectors. However, the characteristics of the selected material(s) (such as capacitive impedance, resistive loss, and physical dimensions) will directly impact the overall maximum length that can be achieved. The System Designer must verify that the signal degradation for the selected medium configuration will meet the constraints set by the electrical and timing specifications given in ***Section 6*** of the I3C Specification ***[MIPI06]***.

5.6.2.3 Design/Matching

The SDA and SCL lines should be matched, both in terms of silicon design and in terms of board design. A mismatch will result in skew, and will limit the timing budget. This is also true for the more complex case of Multi-Lane SDA configurations.

An impedance mismatch between the board and the I/O buffer will result in reflections, directly impacting and reducing the timing window for read/write operations.

For writes, skew is a key parameter that must be controlled for SDR Mode, as well as some HDR Modes. Some HDR Modes perform transfers where the SCL and SDA lines are both being driven. In such cases, Read/Write skew will limit the timing window. This applies to all HDR-TSP/TSL Read transfers, and optionally to HDR-BT Read transfers (i.e., when the Target drives SCL) if allowed by the Controller.

5.6.2.4 GND Management

Although this is not stated in the I3C Specification, in order to ensure good noise immunity performance for reliable operation of single-ended interfaces it is recommended to hold Ground bounce between Controller and Target to below ±50 mV. Given the limited current of the output Driver, the source of this voltage bounce is expected to be parasitic inductances (which can and should be minimized through the use of good PCB layout practices).

5.6.2.5 Stubs

Stubs play a role in maintaining signal integrity. Improperly terminated stubs will result in discontinuity, which in turn will produce reflection and timing loss. The longer the stub, the greater the reflection. The maximum recommended stub delay is 250 ps, which typically corresponds to 1.5" on a FR4 line.

5.6.3 Mixed Bus Considerations

I3C supports Legacy I^2C Devices using Fast-mode (400 KHz) and FastMode+ (1 MHz) with the 50 ns Spike Filter, but no other I^2C modes, nor Legacy I^2C Devices that stretch the clock.

Additionally, if any Legacy I^2C Devices with Index 2 (i.e., that are unable to tolerate higher-speed transfers and do not have the 50 ns Spike Filter) are present on the Bus, then it is a Mixed Slow/Limited Bus (per I3C Specification ***Section 5.1.2.4 [MIPI06]***), so the Controller will be unable to utilize the I3C Bus for higher-speed transfers. Additionally, because the Index 2 I^2C Devices would not properly ignore HDR Mode transfers, the HDR Modes will also be unavailable.

5.6.4 Hot-Join Capability

The I3C Bus protocol supports a Hot-Join mechanism which allows Target Devices to join the I3C Bus after it has already been configured. To ensure stability when a Target Device joins the I3C Bus, the I3C Specification ***[MIPI06]*** defines the conditions under which a Target Device can issue a Hot-Join Request: the Target must first know (or be informed) that the Bus is an I3C Bus, and is then required to wait for a Bus Idle condition.

Note:

> *For extensive detail regarding the Hot-Join feature, Hot-Plug, and related topics, see the separate* ***MIPI Alliance I3C Application Note: Hot-Join [MIPI09]*** *which expands on the Hot-Join section that appeared in earlier versions of this Application Note.*

If the System Designer plans to support Hot-Joining Devices (which might also be Hot-Plug Devices), then to protect the Bus, care must be taken to ensure that I/O pads are safe when unpowered, and that any stub lengths are factored into the overall topology (both before and after the Hot-Joining Devices are connected, powered, and ready to act as I3C Targets).

5.7 Physical/Electrical/Testing Considerations (I3C CTS)

In order to improve interoperability of products implementing the I3C interface, the MIPI Alliance I3C Working Group has developed a Conformance Test Suite (CTS) for I3C and I3C Basic ***[MIPI10]***.

This CTS contains tests designed to determine whether a product conforms to a subset of the requirements defined in the latest versions of either the I3C Specification ***[MIPI03]*** or the I3C Basic Specification ***[MIPI05]***.

In order to promote interoperability testing for I3C Devices in the marketplace, this version of the CTS provides a thorough set of tests that focus on I3C's essential capabilities:

- SDR-only Devices without optional I3C capabilities
- All Error Detection and Recovery methods (for both Controller Devices and Target Devices)
- Basic HDR Enter/tolerance/Restart/Exit

The I3C WG plans to continue expanding the scope of the CTS through future revisions, eventually encompassing all required and optional features of the I3C Specification.

The CTS is organized as one section of tests for a Controller device under test (DUT), and a separate section of tests for a Target DUT. Within each section, the tests are correlated with the I3C Specification Sections containing the related requirement(s), to make it easier to find the relevant Specification details.

5.8 Bus High-Keeper

The I3C Bus always needs a weak High-Keeper Pull-Up on the SDA line, for times when there is no active drive and no strong Pull-Up resistor (or equivalent). The High-Keeper Pull-Up requirements are defined at ***Section 5.1.3.1*** of the I3C Specification ***[MIPI06]***.

Per the I3C Specification, this High-Keeper Pull-Up can be provided in either of two ways:

1. The I3C Controller can be responsible for this, providing the Bus High-Keeper in some appropriate manner.

   Note:

   If the Bus has multiple Controller-capable Devices (i.e., if there are any Secondary Controllers), then each one will also need to be able to provide the High-Keeper while it is the Active Controller. If any Secondary Controller cannot provide a High-Keeper while holding the Controller Role, then the second method below would need to be used. However, a Secondary Controller should disengage its High-Keeper whenever it is not the Active Controller (and especially after it has passed the Controller Role to another Controller-capable Device).

2. When not supported by the Active Controller (or not supported by all Controller-capable Devices), the Bus must be wired with one or more weak passive Pull-Up resistors on the SDA line. These could be 50 KΩ, 100 KΩ, or higher-value resistors. The exact value, and whether more than one resistor is used in parallel, depends on the particular system design, in terms of leakage sources (e.g., the number of Target Devices and Controller Devices over the Bus topology) and the system's noise induction characteristics.

   Note:

   If the Active Controller will not be using its strong (i.e., Open Drain class) Pull-Up on the SDA line during the Bus Free condition or any other longer condition (i.e., the Bus Available condition or the Bus Idle condition), then a stronger passive Pull-Up resistor should be used. This will prevent noise from causing false START requests.

Additionally, HDR-BT Mode uses a "Park1,High-Z" convention for certain protocol elements (such as Transition Bytes, per the I3C Specification at ***Section 5.2.4.2 [MIPI06]***) that require the SDA line to briefly be in a High-Z state. This might require the Controller to momentarily disengage its High-Keeper (if it is capable of doing so), otherwise the System Designer will have to select a sufficiently weak High-Keeper Pull-Up (if wired), so long as the leakage and noise induction concerns listed above are addressed.

Under certain scenarios, the SCL line might also need a weak High-Keeper Pull-Up. If the Active Controller will not be engaging a sufficiently strong Pull-Up on SCL when the Bus is in the Bus Free condition (or other similar conditions where SCL is held High) or when the Active Controller is in deep-sleep state, then a passive weak Pull-Up should be provided on the SCL line. Likewise, if HDR-TSP Mode or HDR-TSL Mode will be used and the Active Controller will not be providing a weak High-Keeper Pull-Up on SCL during the handoff procedure (i.e., Bus Turnaround to a Target for a Read), then a weak passive Pull-Up must be wired onto SCL.

Note:

For the High-Keeper Pull-Up on the SCL line, the exact value, and whether more than one resistor is used in parallel, depends on the particular system design. While the considerations will be similar to those used for the SDA line, leakage sources on the Bus will likely be far less than for the SDA line, since the SCL line is passive for most Target Devices, and the incidence of noise on the SCL line is less likely. It is important that this passive High-Keeper Pull-Up is strong enough to hold SCL High during a long-idled Bus state.

Example: If the total leakage current is determined to be 10 µA from the I3C Devices on the SDA, and we are using a 3.3V Bus, then the minimum resistance needed is:

3.3 V / 0.00001 A = 330 KΩ.

In this example, the System Designer must allow for noise current, which will be lowered by the capacitance of the line; for example, a value of 20 µA displacement can be assumed due to the calculated capacitance of the line flattening a spike of ~2 V ground coupled. The System Designer can simply sum those to equivalent leakages (since it is not typically a concern when noise is in the opposite direction of static leakage, it is only a concern when it is in the same direction)to reach a sum of 110 KΩ. In this example, it is typically safe to use a 100 KΩ Pull-Up resistor.

Further, for a long trace line the System Designer should typically place more than one resistor in parallel along the trace, to ensure stability regardless of the source and placement of the leakage or noise.

Note:

This resistor's only job is to keep the V above V_{IH} (and, in most cases, the difference of $V_{IH} - V_{hys}$) after it is already parked at High (i.e., when it was previously driven High). As a result, a weaker High-Keeper Pull-Up could be used in cases of more noise, as long as the V remains above V_{IH}. The essential goal is to keep V above V_{IH} ***for the I3C Devices on the Bus****, since a V dip along an empty stretch of trace would only matter if it were to impact I3C Devices in other areas.*

5.9 Bridge Devices

The MIPI I3C Specification ***[MIPI06]*** covers inter-Bus bridging support, both passively and actively. In all cases, the bridged endpoints (i.e., the Devices that are being bridged to the I3C Bus) are presented as I3C Virtual Targets, each with its own I3C Dynamic Address, which transact with the I3C Bus through the Bridge Device.

Note:

For extensive detail regarding Bridge Devices, Virtual Targets, and related topics, see the separate ***MIPI Alliance I3C Application Note: Virtual Devices and Virtual Targets [MIPI08]*** *which expands on the Bridge Devices section that appeared in earlier versions of this Application Note.*

This page intentionally left blank.