

embeddedworld
Exhibition&Conference

Demystifying I3C Protocol

Martin Cavallo

10th April 2024

mipi member

binho

Abstract

What?

*"In this presentation, we will **share the expert insights** we have **acquired at Binho** while **working with the I3C protocol.**"*

Why?

*"The session aims to **demystify the protocol and help engineers** overcome all the barriers they might face in their first steps with **I3C adoption**, (...)"*

How?

*"(...) **showing how to get up and running with the protocol** as quickly as possible using hardware and test tools currently available."*

Agenda

- ~~What is MIPI I3C protocol?~~
1. What hardware setup and tools we need?
2. What about MIPI I3C version?
3. Getting to know our devices
4. How fast can I3C run?
5. Dynamic address assignment
6. I3C Private transfers & CCCs
7. In-Band Interrupts & Hot Join

1. Hardware setup and debugging tools

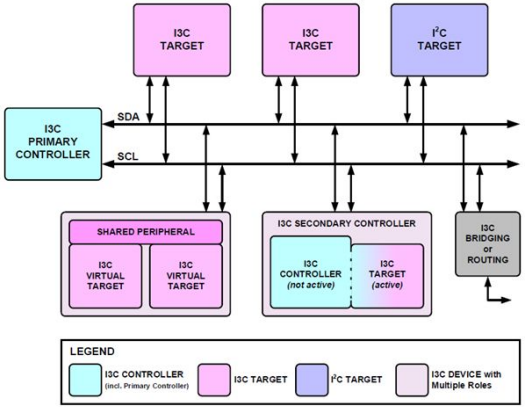
I3C bus basic setup



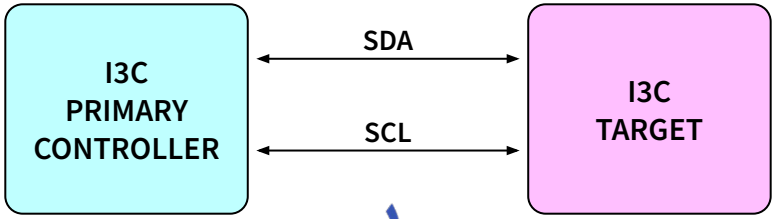
USB Host Adapters



Dev & Eval boards



Point-to-Point Bus Topology



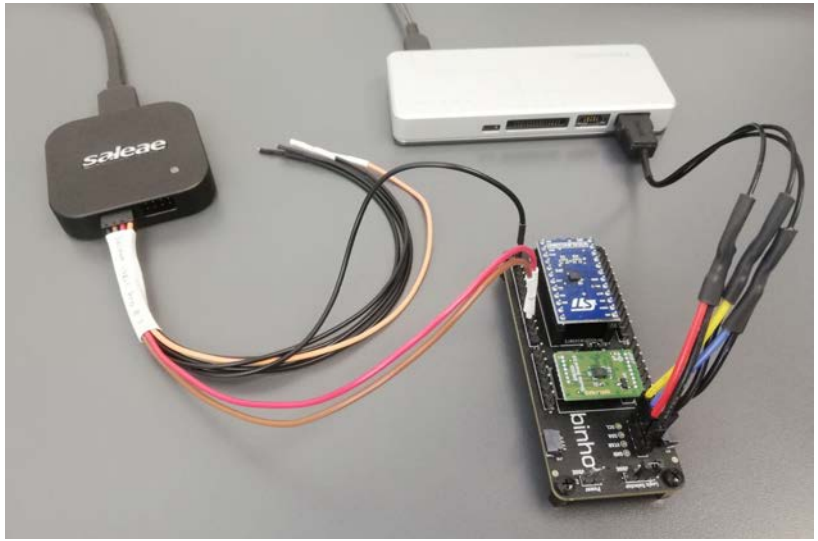


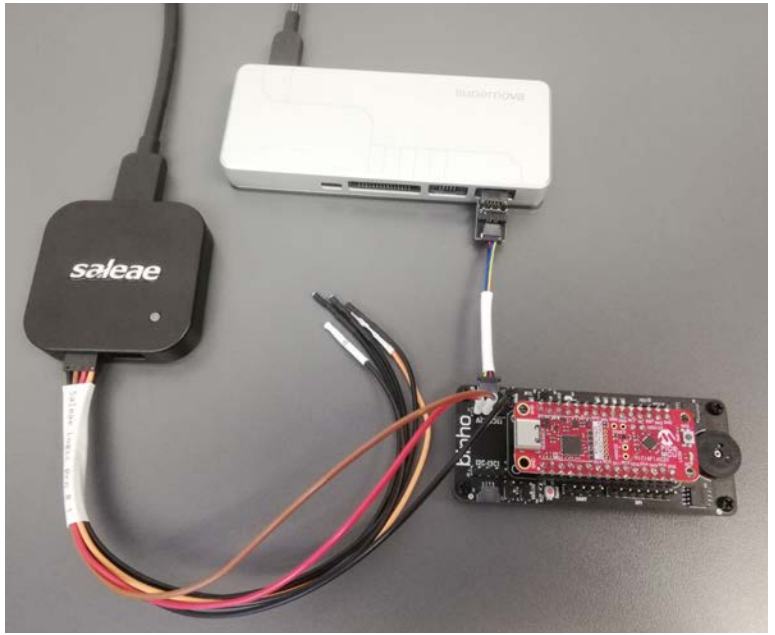
Logic Analyzer & Decoder

Dev & Eval boards



I3C bus basic setup



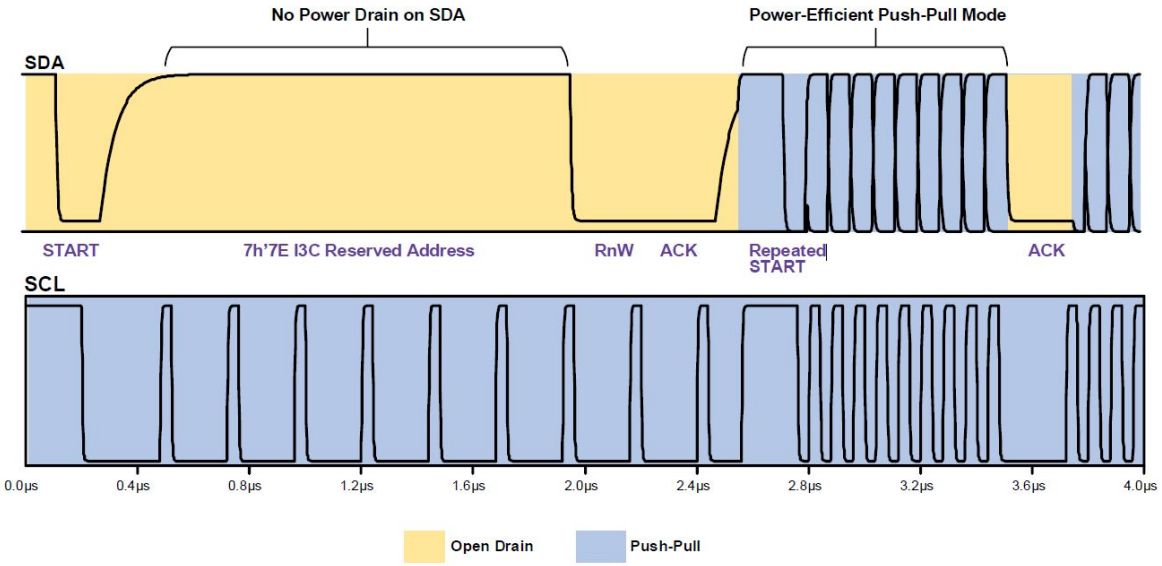


Does I3C require Pull-Up resistors on the bus like I²C?

Only for SDA. Needed for transfers Open-Drain stages and optionally I2C backward compatibility depending on the bus type. Managed by the Active Controller.

What about SCL?

SCL should always be driven by the Active Controller in Push-Pull.



Should we provide an external Pull-Up resistor for SDA?

Not necessarily. Could either be provided internally by the Controller, or it could be an external device/resistor controlled by a Controller output pin (e.g: NXP LPC5536).

Anything else?

A High-Keeper (weak Pull-Up) for both SDA and SCL required for Controller-to-Target and Target-to-Controller Bus handoff, as well as optionally when the bus is idle.

What about I3C Bus voltage?

The **I3C interface** targets three nominal operating voltages: **1.2V, 1.8V, 3.3V**.

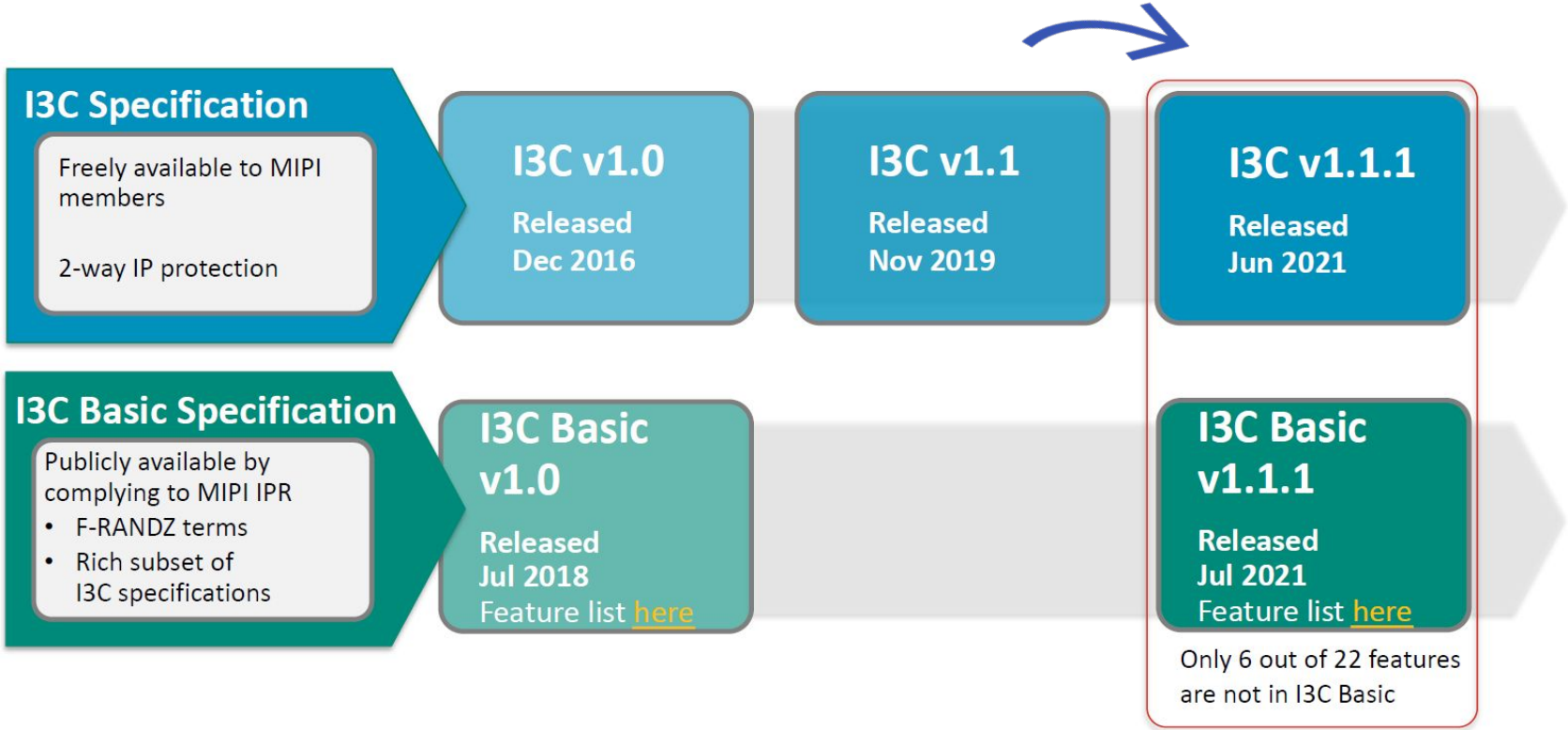
Optionally, the **I3C Basic interface** also targets a nominal operating voltage of **1.0V** and a 100 pF capacitance load (e.g: TI TMP139 sensor).

The I3C interface is not characterized for 5V systems, but could be extended to support 5V if sufficient driver strength, reduced diameter, and/or reduced speed is used in the system.

2. MIPI I3C version

MIPI I3C Specification

Pure editorial update: No new features or capabilities, resolves issues, clarifies, and improves on the I3C v1.1 specification.



How can I know the I3C version supported by the target?

1. Refer to device datasheet. Unfortunately, not all the datasheets include this information. Don't give up!

2. **The I3C Controller must send the GETHDRCAP (v1.0)/GETCAPS(v1.1+) CCC and check how many bytes the target returns.**

   The formerly optional CCC GETHDRCAP has been changed to GETCAPS in I3C v1.1, and its support is required for I3C v1.1 Devices, so as to indicate it is a v1.1 (or later) Device.

3. Optionally, if the I3C Controller is compatible with I3C v1.0, it might send the Direct RSTDAA and check if the target ACKs or NACKs the command.

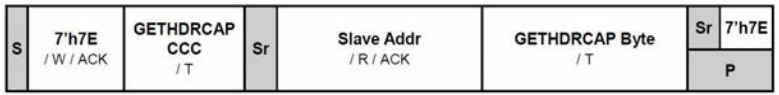
   The Directed form of the RSTDAA CCC is not supported in I3C v1.1 and beyond.

GETHDRCAP (v1.0) vs. GETCAPS (v1.1+)

GETHDRCAP required only if the target supports any HDR mode.

I3C Targets that support I3C version 1.0 return just a **single byte**.

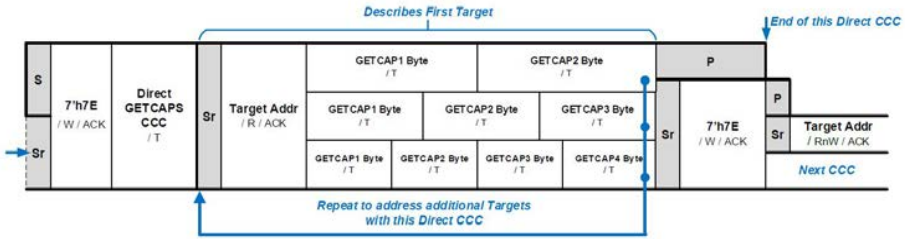
An I3C Controller that supports I3C v1.1 must recognize a valid scenario.



GETHDRCAP Format

GETCAPS Format 1 required for all I3C Targets.

All I3C Targets that support I3C version 1.1 or greater shall return **at least** the first **2 bytes**.

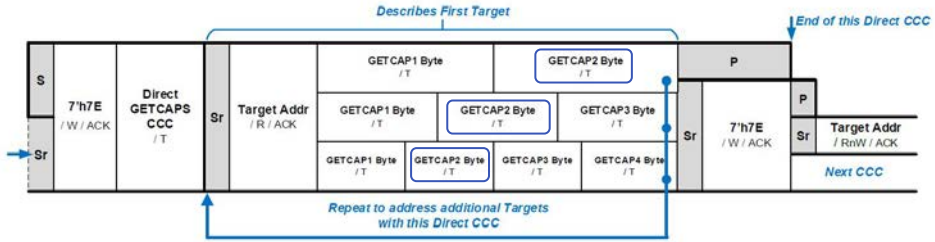


GETCAPS Format 1

GETCAPS CCC and future versions

Bits	Field	Description
7	**HDR-DDR Abort CRC**	Is this I3C Target capable of emitting the CRC Word when a transaction in HDR-DDR Mode is aborted? **0:** No **1:** Yes
6	**HDR-DDR Write Abort**	Is this I3C Target capable of issuing the Write Abort in HDR-DDR Mode? **0:** No **1:** Yes
5:4	**Group Address Capabilities**	This 2-bit integer indicates the Group Address function capabilities of this I3C Device. **Values:** **0:** Does not support Group Address function **1:** Can be assigned one Group Address **2:** Can be assigned two Group Addresses **3:** Can be assigned three or more Group Addresses
3:0	**I3C 1.x Specification Version**	This 4-bit integer indicates the minor version number of the MIPI I3C Specification with which this I3C v1.x Device complies. That is, the 'x' in "I3C v1.x". **Examples:** If Bits[3:0] are set to 4'b0001 (decimal 1), then the Device complies with I3C v1.1 (or v1.1.1, etc.) If Bits[3:0] are set to 4'b0010 (decimal 2), then the Device complies with I3C v1.2 (Note: Bits[3:0] set to 4'b0000 is an illegal value.)

GETCAPS2 Byte Format



GETCAPS Format 1

3. Getting to know the targets

What I3C features and capabilities my target supports?

1. Get the **Bus Characteristics Register (BCR)**.

   *Direct **GETBCR** CCC is a Get request for one I3C Target Device to return its Bus Characteristics Register (BCR) to the Controller.*

2. Get **Optional Feature Capabilites**.

   *Direct **GETCAPS** CCC allows the Controller to query an I3C Target to determine what optional I3C features that Target supports.*

3. Get the **Device Characteristics Register (DCR)**.

   *Direct **GETDCR** CCC is a Get request for one I3C Target Device to return its Device Characteristics Register (DCR) to the Controller.*

4. **Optionally**, get **Max Data Speed**.

   *This conditional Direct **GETMXDS** CCC allows the Controller to determine the SDR Mode data speed limitations of one Target Device if Bit[0] of the BCR is set.*

1. Bus Characteristics Register or BCR

8-bit read-only register that describes target characteristics:

- 🔧**Role:** I3C Target, I3C Controller-capable.
- 🔧**Advanced capabilities:** See **GETCAPS.**
- 🔧**Virtual Target Support:** it is or exposes other downstream devices.
- 🔧**Offline capabilities:** capable of retain the dynamic address in an offline state.
- 🔧**In-Band Interrupts** and payload support.
- 🔧**Max Data Speed Limitation:** It has speed limitations or not.

⚠️It is required to identify the I3C version first!⚠️

BIT	Name	Description
BCR[7]	**Device Role [1]**	**2'b00:** I3C Slave **2'b01:** I3C Master **2'b10:** Reserved for future definition by MIPI Sensor WG **2'b11:** Reserved for future definition by MIPI Sensor WG
BCR[6]	**Device Role [0]**	
BCR[5]	**SDR Only / SDR and HDR Capable**	**0:** SDR only **1:** HDR capable. Use **GETHDRCAP** CCC to determine wich mode(s).
BCR[4]	**Bridge Identifier**	**0:** Not a Bridge Device **1:** Is a Bridge Device
BCR[3]	**Offline Capable**	**0:** Device will always respond to I3C Bus commands **1:** Device will not always respond to I3C Bus commands
BCR[2]	**IBI Payload**	**0:** No data byte follows the accepted IBI **1:** Mandatory one or more data bytes follow the accepted IBI. Data byte continuation is indicated by T-Bit.
BCR[1]	**IBI Request Capable**	**0:** Not Capable **1:** Capable
BCR[0]	**Max Data Speed Limitation**	**0:** No Limitation **1:** Limitation. Use **GETMXDS** CCC to interrogate the Target for specific limitation.

I3C v1.0

BIT	Name	Description
BCR[7]	**Device Role [1]**	**2'b00:** I3C Target **2'b01:** I3C Controller-capable **2'b10:** Reserved for future definition by MIPI Alliance I3C WG **2'b11:** Reserved for future definition by MIPI Alliance I3C WG
BCR[6]	**Device Role [0]**	
BCR[5]	**Advanced Capabilities**	**1:** Supports optional advanced capabilities. Use **GETCAPS** CCC to determine which ones. **0:** Does not support optional advanced capabilities.
BCR[4]	**Virtual Target Support**	**0:** Is not a Virtual Target and does not expose other downstream Device(s). **1:** Is a Virtual Target, or exposes other downstream Device(s).
BCR[3]	**Offline Capable**	**0:** Device will always respond to I3C Bus commands. **1:** Device will not always respond to I3C Bus commands.
BCR[2]	**IBI Payload**	**0:** No data byte follows the accepted IBI **1:** One data byte (MDB) shall follow the accepted IBI, and additional data bytes may follow.
BCR[1]	**IBI Request Capable**	**0:** Not Capable **1:** Capable
BCR[0]	**Max Data Speed Limitation**	**0:** No Limitation **1:** Limitation. Use **GETMXDS** CCC to interrogate the Target for specific limitation.

I3C v1.1

2. Get Optional Feature Capabilities - GETCAPS CCC

If BCR bit [5] is set, the I3C Target might support other optional advanced capabilities:

- 🔧**HDR Modes:** if the target supports any of the HDR modes
- 🔧**Group Address:** the Controller can group targets with a single address.
- 🔧**In-Band Interrupts for Pending Read Notification:** notify the Controller of pending reads.
- 🔧**Defining Bytes** in GETCAPS and GETSTATUS CCCs
- 🔧**Device-To-Device Tunneling (D2DT):** allows two targets to exchange data directly.
- 🔧**Multi-Lane Data Transfer:** support for DUAL (2x SDA) or QUAD (4x SDA) Lane transfers.

3. Device Characteristics Register or DCR

8-bit read-only register that describes the I3C compliant Device type.

255 available codes for describing the type of sensor or device.

MIPI Alliance maintains a public registry of approved and pending DCR value assignment known as “Current I3C Device Characteristic Register (DCR) Assignments”,

It can be found on https://www.mipi.org/MIPI_I3C_device_characteristics_register

4. Target Maximum Speed Limitation

Move on to the next question.

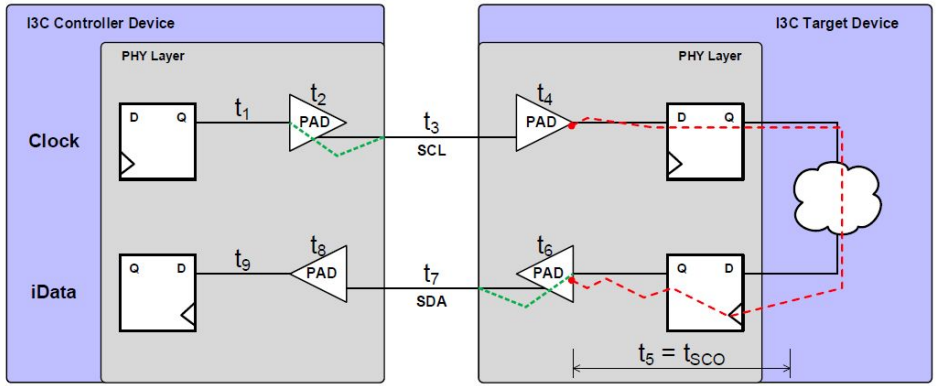
4. How fast can I3C run?

I3C Target speed limitation

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** (max. I3C SDR frequency)
- **Clock-to-Data Turnaround Time (t_{sco}) greater than 12 ns**.
- Overall **internal Device delays**, including Pads, is greater than 12 ns.
- The **Device requires more** time to prepare the requested data.

As a result, the Controller should send a **GETMXDS** command.



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

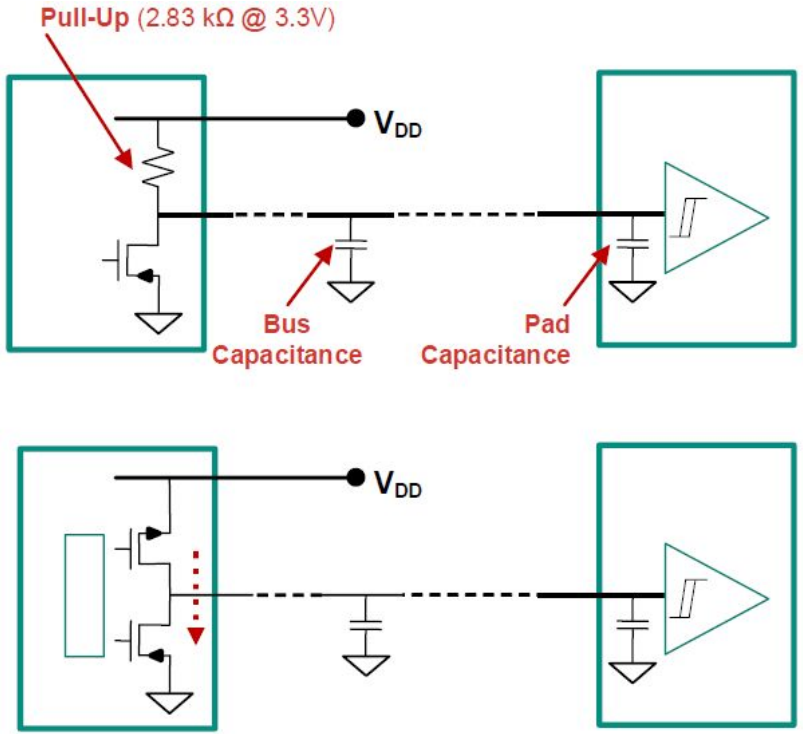
Hardware design constraints

I3C Total Bus Capacitance

It is the sum of the **bus capacitance** and the input/output **pads capacitance** of the devices.

In I3C the assumed **total bus capacitance is below 50 pF**.

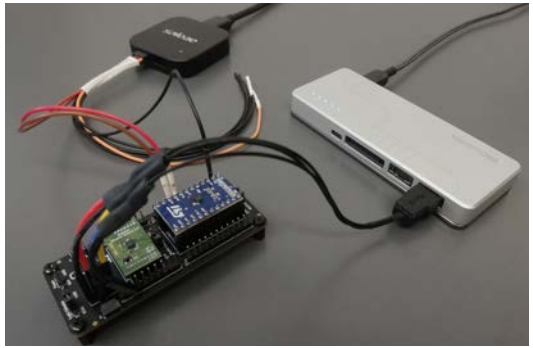
🚨 Keep the bus capacitance under control!

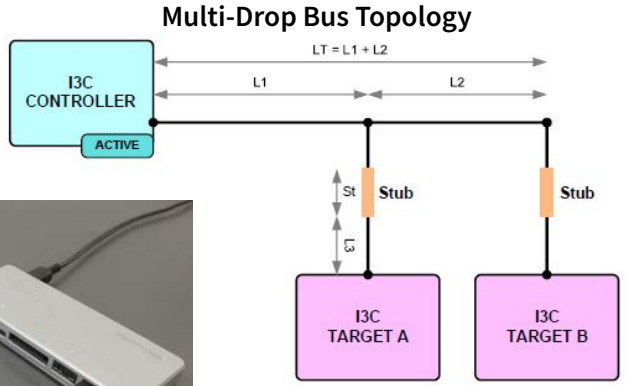


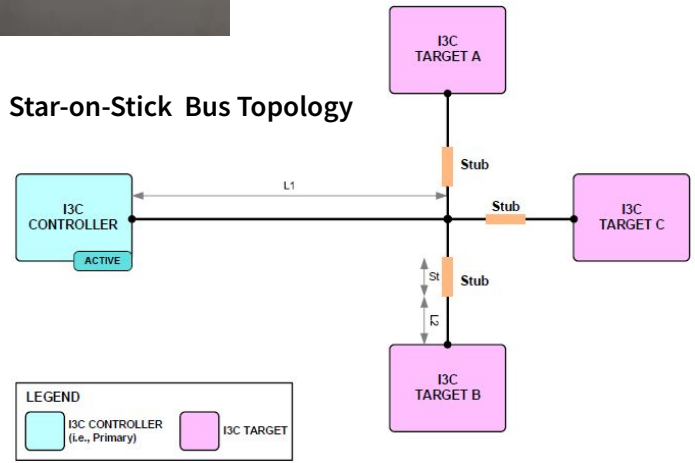
Hardware design constraints

I3C Bus Capacitance depends on:

- Wire and traces length
- Bus topology:
  - Point-to-Point Bus Topology
  - Multi-Drop Bus Topology
  - Start-on-Stick Bus Topology
- Medium and Traces:
  - board trace
  - board trace + via + trace + cable
  - stubs and reflections
- SDA and SCL design matching







5. Dynamic address assignment

Address Assignment

💡**Think out-of-the-box!** The Dynamic Address Assignment **relies on the application**.

⚠️**Caution!** There are reserved and forbiden addresses that must not be used.

👩‍💻**Management required!** The Controller must control the status of every address value.

- ✔️**Invalid** address use
- ✔️**Addresses of I2C devices** connected to the I3C bus (mixed bus)
- ✔️Status of valid and available addresses for I3C: **Assigned, Not Assigned**

📂**Storage required!** Save targets information in the **Target Devices Table.**

🛴**The mean!** Chose the CCC according to the information provided:

ENTDAA, SETDASA, SETAASA

🔝Set **priorities for In-Band Interrupt requests.** Lowest address, highest priority.

Target Devices Table (TDT)

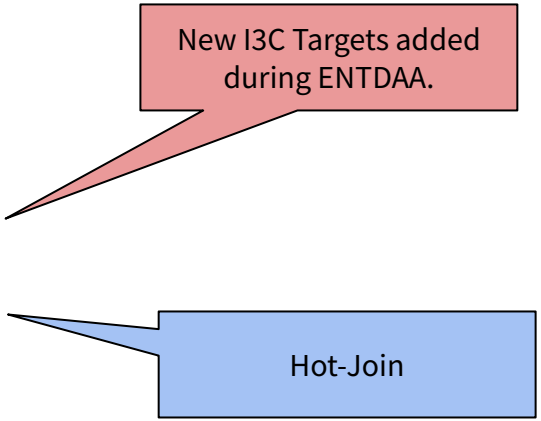
Index	Dynamic Address	Static Address	BCR	DCR	PID	Type	Assignment	IBIs
0	0x08	-	0xEF	0xC6	0x01 0x02 0x03 0x04 0x05 0x06	I3C	ENTDAA	ACCEPT
1	0x09	0x69	-	-	-	I3C	SETDASA	REJECT
2	-	0x50	-	-	-	I2C	-	-
3								
4								
5								
6								

I3C Targets can be added before bus initialization. Mainly SETDASA targets whose static address is required.

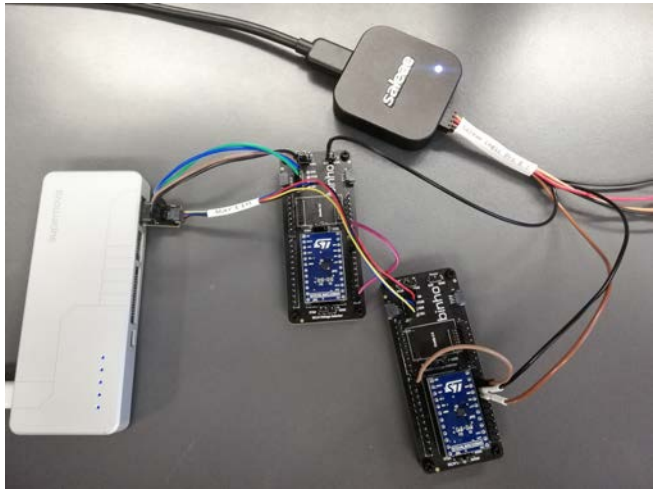
I2C Targets must be added to the table.

Target Devices Table (TDT)

Index	Dynamic Address	Static Address	BCR	DCR	PID	Type	Assignment	IBIs
0	0x08	-	0xEF	0xC6	0x01 0x02 0x03 0x04 0x05 0x06	I3C	ENTDAA	ACCEPT
1	0x09	0x69	-	-	-	I3C	SETDASA	REJECT
2	-	0x50	-	-	-	I2C	-	-
3	0x0C	-	0x0A	0x09	0x34 0x74 0x12 0x0A0x10 0x00	I3C	ENTDAA	-
4	0x0D	-	0x05	0x24	0x10 0x11 0x00 0x22 0x56 0x08	I3C	ENTDAA	-
5	0x0E	-	0x12	0x06	0x02 0x13 0x28 0x01 0x00 0x00	I3C	ENTDAA	-
6								



ENTDAA and PID collision



💡 **One possible approach…**

Set different static address using user-defined address pins SA0-SAn.

Then assign dynamic addresses to the targets throught **SETDASA CCC**.

💡 **Another possible approach…**

Some manufacturers allow to **modify the PID** using also the user-defined address pins A0-An. For instance:

- ST LSM6DSV modifies **Bits[15:12] Instance ID** based on the **SA0** pin.
- NXP P3T1755 modifies **Bits[11:0] vendor-defined** based on **A0-A2** pins.

6. I3C Private Transfers & CCCs

I3C Private Transfers

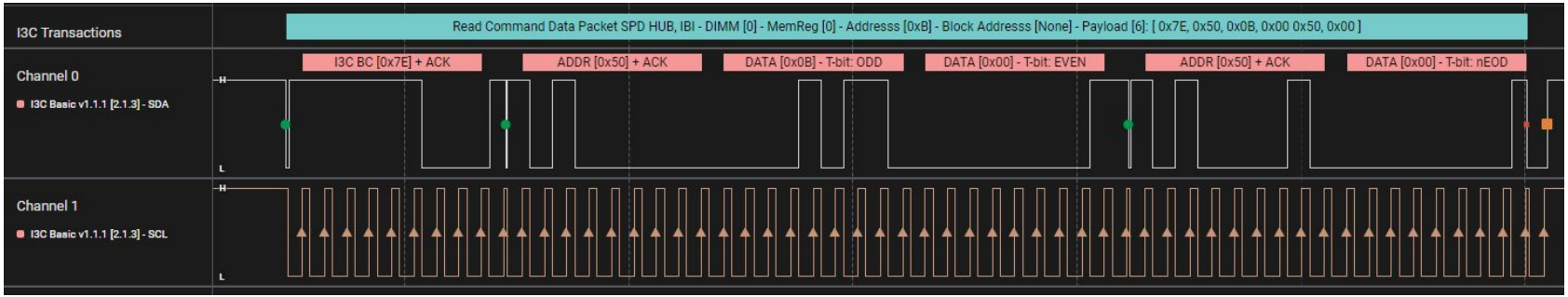
Initialized by the I3C Controller.

Belong to **transport layer**, the data content is **application-specific**. Possible usecases:

🔧Read/Write registers map of a sensor. Registers addressing defined by manufacturer.

🔧Read/Write the pages of a memory. Internal addressing defined by manufacturer.

🔧Transfer application protocols packets: Debug for I3C, MCTP, etc.



I3C Private Transfers

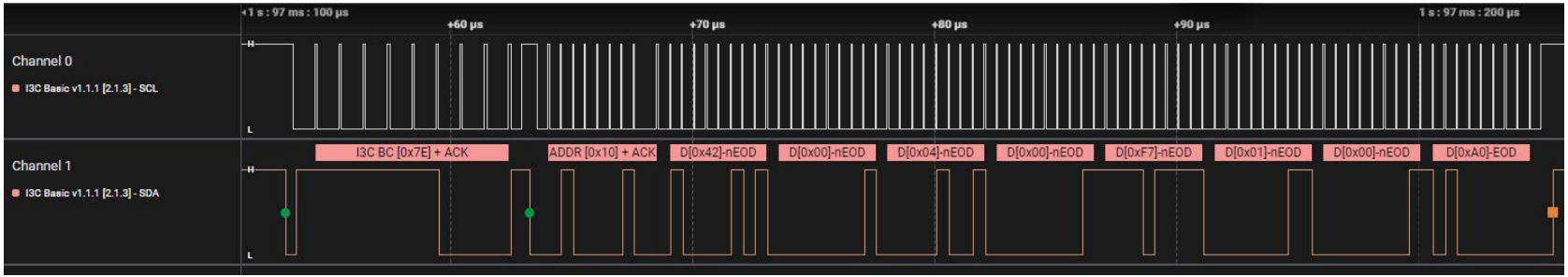
9th bit of the data word, the Transition Bit or **T-Bit**:

❌**vs. I²C**: 1. **Not the ACK** bit 2. **Driven by the transmitter** instead of the receiver.

➡️In data **written by the I3C Controller** means a **Parity bit**, calculated using odd parity.

⬅️In data **written by the I3C Target** means **End-of-Data (EOD):**

- T-Bit = 0 means End of Data. **The Target can end the transfer.**
- T-Bit = 1 means not End of Data.

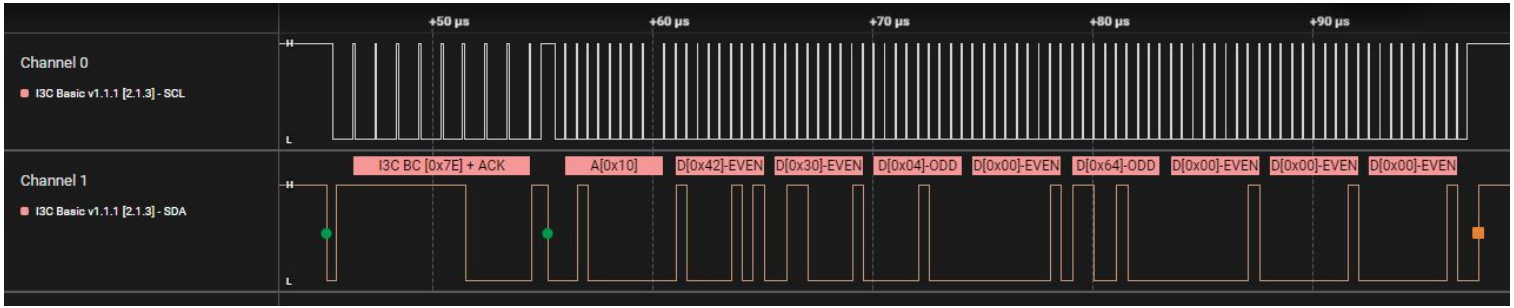


I3C Private Transfers

Upsides of starting the transfer with I3C Broadcast Address header 7’h7E:

➕It is possible to detect the presence of any I3C Targets on the bus and **switch to Push-Pull** mode for a **more efficient transfer**.

➕**Allows any I3C Target to request an In-Band Interrupt** because its address will have higher priority than the Broadcast Address.

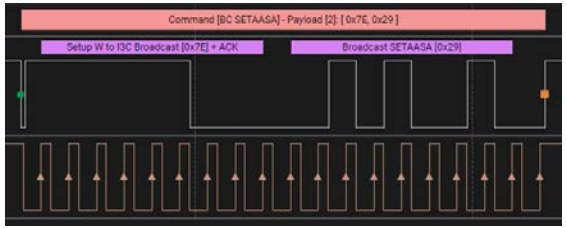


Common Command Codes (CCCs)

Initialized by the I3C Controller, aimed at sending standardized commands to **control the Bus**.

Broadcast CCCs:

- 🔧Addressed to all I3C Targets at the same time.
- 🔧All Broadcast CCCs are Write CCCs.



Direct CCCs:

- 🔧Addressed to a single I3C Target or a group of I3C Targets
- 🔧Three types: Read, Write or Read/Write



7. In-Band Interrupts & Hot Join

In-Band Interrupts

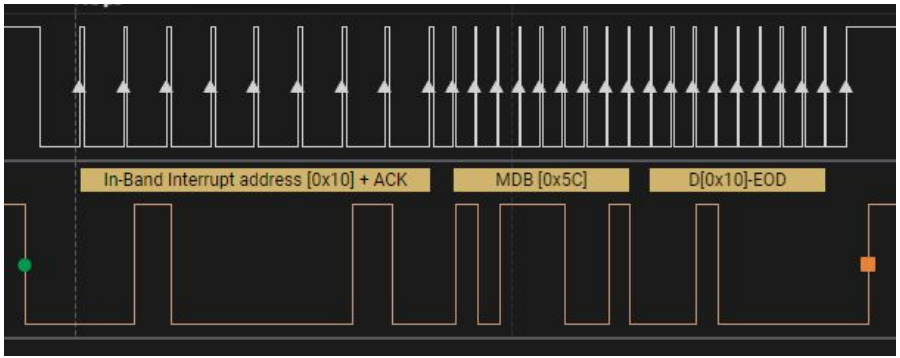
✅Make hardware design clean and simple. Avoid extra out of the band signals.

👨‍💻Require target configuration commonly done through private transfers.

⚠️Don’t forget to enable them using the Direct or Broadcast **ENEC CCC**.

❌The I3C Controller can accept **(ACK)** or reject **(NACK)** the In-Band Interrupt requests based on the TDT.

⛔Disable them using the Direct or Broadcast **DISEC CCC**.



Hot-Join

✅Allows I3C Target join the Bus after it is already configured.

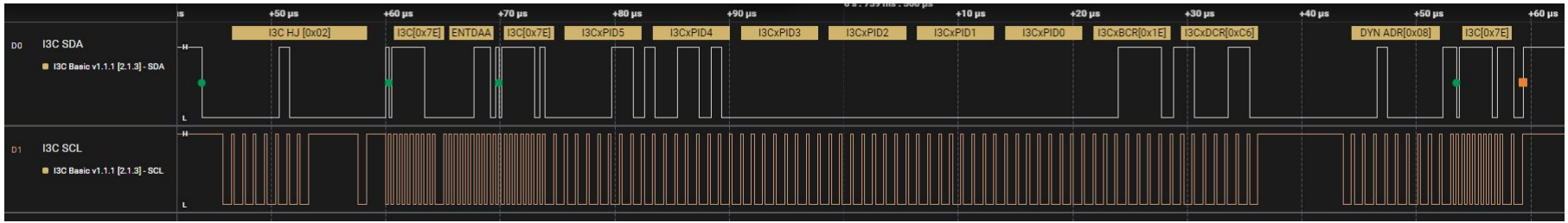
✅A special type of IBI with reserved address **7’h02/W**. Highest priority level.

✅Recommended for low-power applications.

⚠️Don’t forget to enable it using the Direct or Broadcast **ENEC CCC**.

❌The I3C Controller can accept **(ACK)** or reject **(NACK)** the HJ.

⛔Disable HJ requests using the Direct or Broadcast **DISEC CCC**.



Thank you!

Martin Cavallo
martin@binho.io

Visit us on **binho.io**

mipi member

binho