binho

APPLICATION NOTE

Moving a bus from SMBus at 3.3 V to I3C at 1.8 V with the Binho Supernova

AN0018

Document information

Keywords	AN0018, Supernova, I3C, SMBus, 7Eh, DISEC, SETDASA, ENTDAA, dynamic address assignment, bus voltage, pull-up, NVMe-MI, pyCosmicSDK
Abstract	A device that starts on an SMBus bus at 3.3 V and moves to I3C at 1.8 V needs the host to find it, check that nothing SMBus-only is left on the bus, lower the bus voltage and assign it a dynamic address. The Supernova does all of this from one I3C port. This note supplies a script that runs the five steps, with SETDASA or ENTDAA for the last one, and shows it on a bench where a second Supernova plays the I3C device: 20 of 20 runs with SETDASA and 20 of 20 with ENTDAA, with the bus high level measured at 3.34 V before the change and 1.79 V after it.

Binho Inc.
42 Dore Street, Unit A, San Francisco, CA 94103

REVISION 1.0
6 NOVEMBER 2026

1 Introduction

Some devices power up as SMBus targets at 3.3 V and can move to I3C at 1.8 V. The host decides when: it first makes sure that every device left on the bus can follow, then lowers the bus voltage and assigns dynamic addresses. A device that only speaks SMBus would not survive the move, so the host stays in SMBus mode if it finds one.

The Supernova's I3C port speaks SMBus (legacy I2C) and I3C on the same two pins, and its voltage is set from software. The whole sequence therefore runs from one port and one script.

1.1 Scope

This document covers:

- the five steps of the sequence and the pyCosmicSDK call behind each one
- the wiring, and where the bus pull-ups must go so that they follow the voltage change
- a script that runs the sequence, with SETDASA or ENTDAA as the address assignment
- measured results on a bench with a Supernova as the I3C device

It does not cover the SMBus to I3C Basic detection flow of the PCI-SIG out-of-band management specification (Hot-Join, ARP, the SMBus reset), which AN0014 covers, or MCTP and NVMe-MI messages.

1.2 Applicable devices

Table 1. Applicable devices

Role	Part	Identifier	Basis
Controller	Binho Supernova	hardware revision C, firmware 4.6.2, I3C HV port	measured, this document
Controller	Binho Supernova	firmware 4.6.0 or later	expected: every call used is in 4.6.0
I3C device	Binho Supernova as I3C target	hardware revision C, firmware 4.6.2	measured, this document
I3C device	any device that answers on SMBus at a static address and supports SETDASA or ENTDAA		not measured

1.3 Related documents

Table 2. Related documents

Document	Title
MIPI I3C	MIPI Alliance Specification for I3C, version 1.1.1
SMBus 3.2	System Management Bus Specification, version 3.2
AN0014	Single-port I2C/SMBus/I3C operation for NVMe-MI on Binho Supernova

2 System overview

Table 3 lists the steps and the call that performs each.

Table 3. Steps of the sequence

Step	Action	pyCosmicSDK call
1	Operate in SMBus mode at 3.3 V	`set_voltage(voltage_mv=3300)`, `bring_up(...)`, `ccc.rstdaa()`
2	Broadcast a 7Eh command. An ACK means at least one I3C device is on the bus	`ccc.disec_broadcast(I3cEvent.HJ)`; a NACK raises `I3cError` with `FW_I3C_NACK_ADDRESS`
3	Probe every SMBus address. A device that answers and is not a known I3C device is SMBus-only; if one remains, stay in SMBus mode	`legacy_i2c_read(address, 1)` for 08h to 77h
4	Move the bus and its pull-ups from 3.3 V to 1.8 V	`set_voltage(voltage_mv=1800)`
5	Assign dynamic addresses with SETDASA or ENTDAA, then read each device's PID	`ccc.setdasa(static=, dynamic=)` or `init_bus()`, then `ccc.getpid(address)`

Three facts shape the sequence:

- **Step 2 is a CCC, not an SMBus transfer.** Only the 7Eh header is sent open-drain, so an SMBus-only device sees an address it does not own and ignores it. The CCC used is DISEC with DISHJ, which also keeps the I3C devices from Hot-Joining while the host is still in SMBus mode.
- **Step 3 needs a list of the I3C devices' static addresses.** An I3C device with a static address also answers an SMBus read at that address, so an answer alone does not prove a device is SMBus-only. The script takes the I3C devices' static addresses (`--i3c-static`) and treats any other answer as SMBus-only.
- **Step 4 moves the pull-ups only if they hang from the port voltage.** The Supernova's SDA pull-up follows the I3C port voltage. An external pull-up follows it only if it is tied to the

port's VTARG pin (section 3.3).

3 Hardware setup

3.1 Required equipment

Table 4. Required equipment

Item
Binho Supernova, firmware 4.6.0 or later
USB-C cable
The device under test, or a second Supernova to play it (section 6)
Pull-up resistors to VTARG, if the bus has none

3.2 Connections

Table 5. Connections between the Supernova I3C HV port and the device

Supernova I3C HV port	Device	Required
SCL	SCL	required
SDA	SDA	required
GND	GND	required
VTARG	pull-up resistors, or the device I/O supply	required if the bus has external pull-ups

3.3 Bus considerations

The I3C port has no pull-up on SCL, and SMBus needs one. Tie the SMBus pull-ups to the port's VTARG pin, not to a fixed 3.3 V rail. VTARG follows `set_voltage`, so step 4 takes the pull-ups from 3.3 V to 1.8 V with the bus. A pull-up to a fixed 3.3 V rail would hold the bus at 3.3 V after step 4, above what a 1.8 V device accepts.

> **Caution:** a device that is not 3.3 V tolerant must not be on the bus during steps 1 to 3.
>
> **Caution:** only one controller on the bus.

4 Software setup

Install the latest pyCosmicSDK release from PyPI with the same Python that runs the script:

```
python -m pip install --upgrade pycosmicsdk
```

Update the Supernova to firmware 4.6.0 or later with Mission Control.

5 Procedure

Run the script with the static address of each I3C device on the bus. The defaults are 3300 mV for SMBus, 1800 mV for I3C, and SETDASA with dynamic addresses from 08h:

```
python smbus_to_i3c_daa.py --i3c-static 0x3A --daa setdasa
python smbus_to_i3c_daa.py --i3c-static 0x3A --daa entdaa
```

The output names each step. On the bench of section 6:

```
FW-4.6.2 controller | target FW-4.6.2
1. SMBus mode, bus at 3300 mV
2. DISEC to 7Eh: ACK, an I3C device is present
3. SMBus probe 08h-77h: 3Ah -> SMBus-only: none
4. Bus and pull-ups at 1800 mV
5. SETDASA: dynamic 08h, GETPID 0B0000000A01
```

The script stops after step 2 if nothing ACKs 7Eh, and after step 3 if an SMBus-only device answers. In both cases the bus stays at the SMBus voltage. It exits with 0 when every assigned device returned its PID.

To use the sequence in your own code, copy the `run` function: each step is a few lines and uses only the calls in table 3.

Table 6. smbus_to_i3c_daa.py options

Option	Default	Meaning
`--serial`	any	serial number (or prefix) of the controller Supernova
`--i3c-static`	required	static addresses of the I3C devices; any other SMBus answer counts as SMBus-only
`--daa`	`setdasa`	`setdasa` assigns one address per static address; `entdaa` lets the firmware assign every device
`--first-dynamic`	0x08	first dynamic address that SETDASA assigns
`--smbus-mv`	3300	bus voltage for steps 1 to 3
`--i3c-mv`	1800	bus voltage, from step 4 on
`--settle-ms`	10	wait after the voltage change
`--emulate-target`	none	serial of a second Supernova that plays the I3C device at the first `--i3c-static` address
`--rounds`	1	run the sequence this many times and print the count that passed

6 Measured results

6.1 Test configuration

Table 7. Test configuration

Item	Value
Controller	Binho Supernova, hardware revision C, firmware 4.6.2 (`FW-4.6.2`), I3C HV port
I3C device	a second Supernova, firmware 4.6.2, as I3C target with static address 3Ah, PID 0B0000000A01, BCR 06h, DCR C5h, via `--emulate-target` ; its port voltage set with the controller's
Host software	pyCosmicSDK 1.6.0 from PyPI, Ubuntu 22.04
Bus	I3C HV ports wired SCL, SDA, GND
Analyzer	Saleae Logic Pro 16, analog and digital on SCL and SDA

6.2 Results

Table 8. Results

Run	Result
SETDASA, 20 runs	20 of 20: 7Eh ACK, only 3Ah on SMBus, dynamic address 08h, PID read back
ENTDAA, 20 runs	20 of 20: same, dynamic address 08h assigned by the firmware
Bus high level, one run on the analyzer	SCL 3.34 V in steps 1 to 3, 1.79 V after step 4

The SMBus-only branch of step 3 was not run on this bench, because it has no SMBus-only device. These are what we measured on one bench, not a specification.

7 Troubleshooting

Table 9. Troubleshooting

Symptom	Probable cause and action
Step 2 reports NACK with an I3C device on the bus	The device is not powered, or not at the SMBus voltage. Check `--smbus-mv` and the device supply.
Step 3 lists the I3C device as SMBus-only	Its static address is missing from `--i3c-static`. Add it.
Step 3 finds nothing, step 5 still works	The device does not answer SMBus reads at its static address. This does not affect the sequence.
Step 5 GETPID fails with `FW_I3C_NACK_ADDRESS`	The bus is still above the device's I/O voltage, or the device does not support the DAA method. Check that the pull-ups are tied to VTARG, or try the other `--daa`.
Bus still at 3.3 V after step 4	External pull-ups to a fixed 3.3 V rail. Tie them to VTARG (section 3.3).

8 Acronyms

Table 10. Acronyms

Term	Meaning
I3C	MIPI Improved Inter-Integrated Circuit
SMBus	System Management Bus
CCC	I3C common command code
DISEC	CCC that disables target events, here Hot-Join (DISHJ)
SETDASA	CCC that gives a device with a static address a dynamic address
ENTDAA	CCC that starts dynamic address assignment for every device
RSTDAA	CCC that resets every dynamic address
GETPID	CCC that reads a device's 48-bit provisional ID
PID	Provisional ID
BCR, DCR	I3C bus and device characteristics registers
VTARG	Supernova I3C port supply output, at the port voltage

9 Note about the source code in the document

Example code shown in this document and supplied in the accompanying archive is provided for evaluation and reference. It is released under the MIT license, the text of which is included in the archive.

The archive `AN0018-assets.zip` contains `smbus_to_i3c_daa.py` and `LICENSE`.

10 Revision history

Table 11. Revision history

Revision	Date	Description
0.1	23 October 2026	Draft for internal review, measured on firmware 4.6.2 and pyCosmicSDK 1.6.0.
1.0	6 November 2026	Published.

Legal information

Definitions

Draft

A document marked draft is under internal review and subject to formal approval. Binho Inc. makes no representation or warranty as to its accuracy or completeness and accepts no liability arising from its use.

Disclaimers

Limited warranty and liability

Information in this document is believed to be accurate and reliable. Binho Inc. makes no representation or warranty, express or implied, as to its accuracy or completeness, accepts no liability for the consequences of its use, and takes no responsibility for content originating outside Binho Inc.

In no event shall Binho Inc. be liable for any indirect, incidental, punitive, special or consequential damages, including without limitation lost profits, lost savings, business interruption, or costs of removing or replacing any product, whether or not based on tort, warranty, breach of contract or any other legal theory. Notwithstanding any damages a customer might incur, the aggregate and cumulative liability of Binho Inc. is limited in accordance with its terms and conditions of commercial sale.

Right to make changes

Binho Inc. reserves the right to change the information in this document, including specifications and product descriptions, at any time and without notice. This document supersedes all information supplied prior to its publication.

Suitability for use

Binho Inc. products are not designed, authorized or warranted for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure of a Binho Inc. product can reasonably be expected to result in personal injury, death, or severe property or environmental damage. Any such inclusion or use is at the customer's own risk and Binho Inc. accepts no liability for it.

Applications

Applications described here are illustrative only. Binho Inc. makes no warranty that they are suitable for the specified use without further testing or modification. Customers are responsible for the design and operation of their own applications and products, for appropriate design and operating safeguards, and for determining whether a Binho Inc. product is suitable and fit for their purpose. Binho Inc. accepts no liability for assistance with applications or customer product design.

Third-party products and specifications

This document refers to products, boards, specifications and documentation supplied by third parties, which remain the responsibility of their respective owners. Binho Inc. makes no warranty regarding them and their behavior may change without notice. Register maps, protocol details and device identifiers reproduced here were observed on the specific hardware and firmware versions stated in this document and must be confirmed against the vendor's own documentation for the reader's part.

Example code

Source code supplied with this document is provided as an example, for evaluation and reference. It is not qualified for production use, has not been developed under a functional safety or security process, and is licensed under the terms accompanying it in the distribution archive.

Terms and conditions of commercial sale

Binho Inc. products are sold subject to the general terms and conditions of commercial sale published by Binho Inc., unless otherwise agreed in a valid written individual agreement, in which case only the terms of that agreement apply.

Export control

This document and the items it describes may be subject to export control regulations. Export may require prior authorization from the competent authorities.

Security

All products may be subject to unidentified vulnerabilities or may support security standards with known limitations. The customer is responsible for the design and operation of its applications and products throughout their lifecycle to reduce the effect of such vulnerabilities, and for compliance with all legal, regulatory and security requirements concerning its products.

Trademarks

All referenced brands, product names, service names and trademarks are the property of their respective owners. **Binho** and the Binho logo are trademarks of Binho Inc. **NVMe** and **NVMe-MI** are trademarks of NVM Express, Inc. **Saleae** and **Logic** are trademarks of Saleae, Inc.