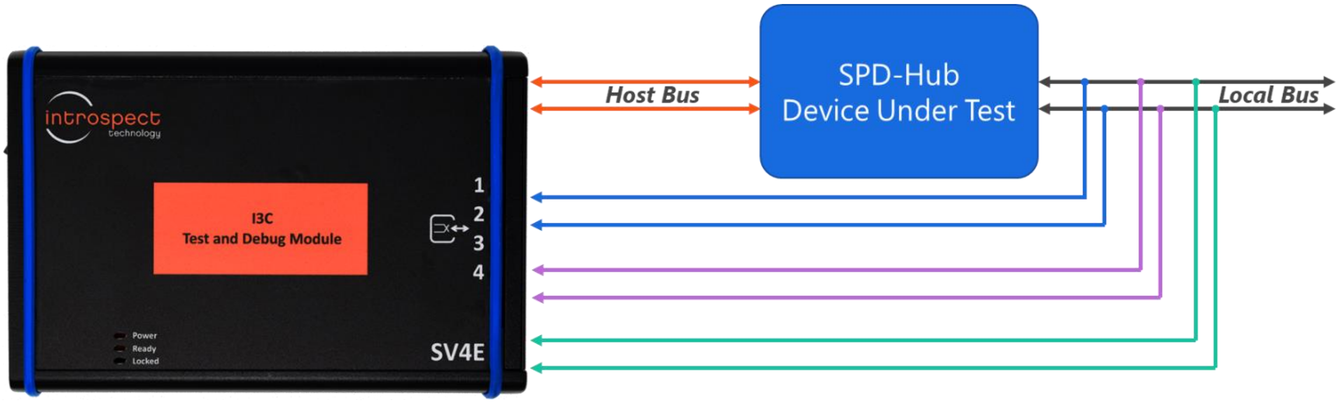
WHITE PAPER

Component Validation of JEDEC® DDR5 SPD Hub Devices

E SERIES



Published in Canada on March 31, 2021
EN-W002E-E-21090

INTROSPECT.CA

Table of Contents

Introduction

Double Data Rate (DDR) technology is a memory chip technology that is pervasively used in server and data center applications as well as in personal computers and gaming stations. This technology has become critical to the deployment of high-performance computers, AI processors, and edge computers in data centers. And now as it enters its fifth (5th) generation in the form of the JEDEC DDR5 standard, it has added new system management buses and components that pose unique development challenges for both component manufacturers and system integrators. Indeed, at the heart of the DDR5 ecosystem is the **SidebandBus Protocol**, which is a specification that governs the behavior of the various system management components and buses that operate a DDR5-based server implementation.

This white paper focuses on the design and validation of the **Serial Presence Detect EEPROM with Hub Function** (SPD5 Hub or simply SPD Hub), which operates using the SidebandBus Protocol. The SPD Hub is crucial to the successful implementation of DDR5 memory modules in particular because the SPD Hub allows for large serial communications buses on large servers to be isolated and broken down into smaller buses. Smaller buses have lower capacitance and can operate at higher frequencies. As such, the SPD Hub sits at the heart of the entire flow for managing the quality and reliability of modern memory modules.

In this white paper, we first introduce the hub architecture in DDR5 and present some of the design challenges associated with it. These include topics like device **address modification** and **packet error codes**. Then, we present a detailed test methodology for validating and characterizing all ports on a SPD Hub device using the **SV4E-I3C I3C Test and Debug Module** from Introspect Technology. Specifically, we illustrate connection diagrams, describe reset and initialization sequences, and detail functional design verification tasks related to reading and writing registers inside the SPD Hub or inside devices behind it.

SPD Hub Device Design and Validation Challenges

SIDEBANDBUS HUB ARCHITECTURE

Figure 1 shows the global SidebandBus hub architecture as defined by the JEDEC JESD403-1 and JESD300-5 specifications. As can be seen, a main controller – such as a CPU in a server implementation – communicates with several devices over a two-wire bus that is based on the MIPI® I3C Basic^{SM} standard with specific modifications that are necessary for JEDEC applications. These modifications are clearly described in the JESD403-1, where different devices classes are introduced. Referring to Figure 1, a device connected to the main controller can be the SPD Hub or it can be a standard I3C device. In any case, the hub architecture consists of a series of hub devices that essentially create their own local buses. Then, peripheral devices reside on these local buses and can communicate with either their corresponding hub or with the main controller through the hub.

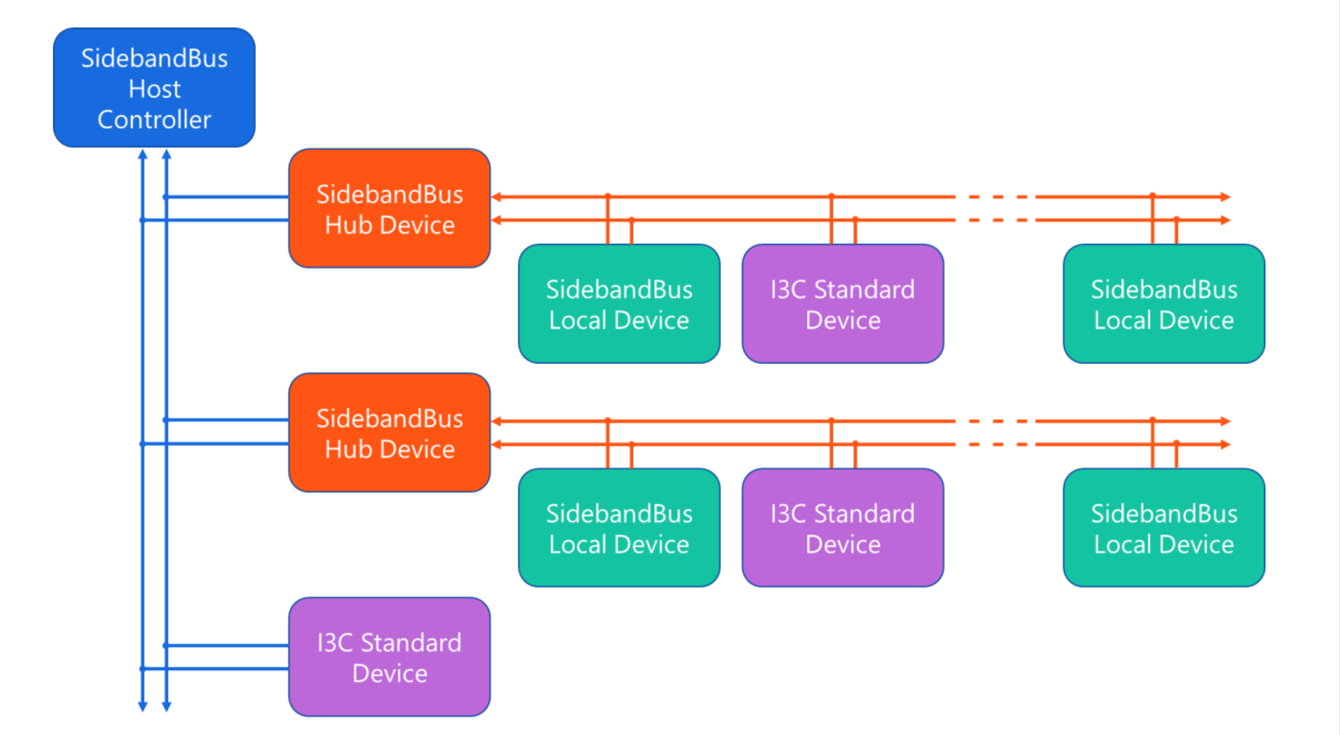


Figure 1: SPD Hub architecture in a DDR5 system implementation.

The hub architecture's main benefit is that it isolates the capacitive load of all local devices from the main SidebandBus Host Controller. This is important especially because the JESD403-1 now enables very high speed communications as will be described later in this document. At the same time, because the architecture still allows the main controller to address all individual peripheral devices, unique design challenges exist. For one, individual addressing of devices means that the SPD Hub device needs to act as both a master and a slave at the same time, and this is illustrated in Figure 2. From an electrical point of view, the Local Bus to the right of the SPD Hub in the figure is completely independent from the Host Bus on the left hand side. Yet, the SPD Hub is still expected to allow the main controller to address devices on the hub's Local Bus without requiring extra address bits, and this is described in the next section.

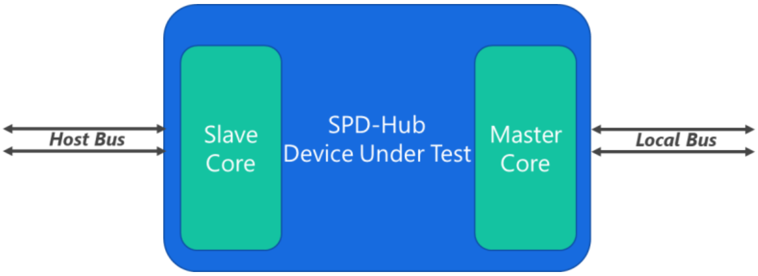


Figure 2: SPD Hub implementation consisting of a slave core and a master core.

ADDRESS MODIFICATION

In order to save on the number of address bits in the JESD403-1 protocol, the SPD Hub architecture allows for reusing the 7-bit addresses on all local devices even when communicating with the main SidebandBus Host Controller. This is achieved using a mechanism called address modification, and it is illustrated at a high level in Figure 3. As can be seen in the figure, the 7-bit I3C address for all devices – including the SPD Hub itself – is divided into two parts:

- 4-bit part with a device type identifier
- 3-bit part that represents the Local Bus on which a device resides

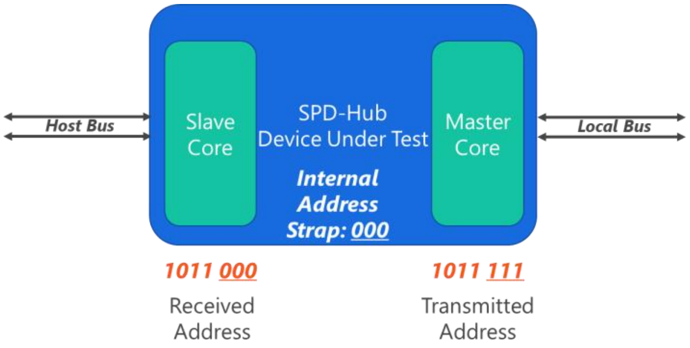


Figure 3: Illustration of the address modification scheme in the SPD Hub when targeting a RCD behind the hub.

Using the above address scheme, a SPD Hub device contains internal mechanisms for setting the 3 least significant bits of its static address. In the example of Figure 3, these bits are set to 000, and this means that a host controller communicating with this hub must specify an address like 1010 000, where 1010 is the device type identifier for the hub and 000 is its internally set address. Now, if the host controller wants to talk to another device behind the hub, like a RCD, then it must specify the address 1011 000, where 1011 is the device type identifier for the RCD. The hub then modifies this address to become 1011 111 when dispatching the communication message (read or write) to the RCD on the Local Bus.

Validating the address modification scheme of the SPD Hub is an important step for ensuring proper operation within a complete server implementation. Later sections in this document will show how the test can be performed.

PACKET ERROR CHECKING

The JESD403-1 protocol supports packet error codes (PEC) in the communication protocol between the host controller and the SPD Hub. These codes are 8-bit words that are transmitted at the end of an I3C transaction, and they represent the CRC value corresponding to the payload data being transmitted. As can be seen, this concept is similar to advanced packet-based communications protocols that are used in telecommunications or other computer buses such as PCI Express. The inclusion of packet error checking is a major addition to the system management function of the SidebandBus protocol, and it represents a significant challenge for design and test.

HIGH FREQUENCY OPERATION

The JESD403-1 protocol now enables operation at a clock frequency of 12.5 MHz. This is more than an order of magnitude increase over previous-generation buses based on I2C. More importantly, the SPD Hub – and the SidebandBus protocol in general – is required to transition its speed from I2C frequencies to 12.5 MHz at very specific times and based on very specific protocol events. Thus, the design of the SPD Hub has to consider all possible combinations closely, especially when a large mix of device types is expected to be attached to the hub.

SPD Hub Component Validation Methods

This section describes a high-level test sequence for validating the functionality and performance of a SPD Hub device. The test sequence is described in the context of the **SV4E-I3C I3C Test and Debug Module**, which is an all-inclusive solution for exercising JESD403-1 buses and for also performing protocol analysis and debug. The ability of the SV4E-I3C to generate compliant and non-compliant traffic and its ability to operate in both slave and master modes both enable the seamless testing of any JESD403-1 device.

BLOCK DIAGRAM OF TEST SETUP

Figure 4 shows a typical bench setup for validating a SPD Hub device under test (DUT). The SV4E-I3C contains 4 dual-mode devices (i.e., devices that can operate as a main master or slave). More importantly, the SV4E-I3C supports up to 4 independent buses. This feature is all-important for SPD Hub testing because it means that a single SV4E-I3C can be connected to the Host Bus side of the SPD Hub as well as the Local Bus side. The only requirement is that the Local Bus side be connected to an independent port on the SV4E-I3C.

Referring to Figure 4, port 1 of the SV4E-I3C can be declared as a main master (emulating a host controller), and it can belong to one bus on the SV4E-I3C. Port 2 can be connected as a slave on the Local Bus side of the SPD Hub DUT. This slave can be programmed to emulate, say, a Thermal Sensor device. Importantly, port 2 belongs to a second bus on the SV4E-I3C and is programmed independently of port 1. Ports 3 and 4 of the SV4E-I3C can also be connected to the Local Bus side of the SPD Hub, and they can be used to emulate PMIC and/or RCD devices. These ports belong to the same second bus on the SV4E-I3C. The availability of these ports provides complete coverage for the SPD Hub's ability to perform address modification, among other things.

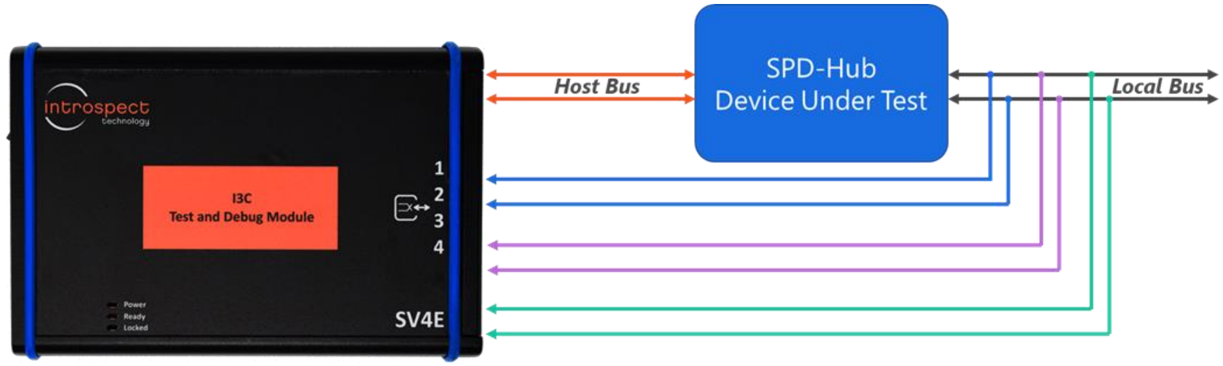
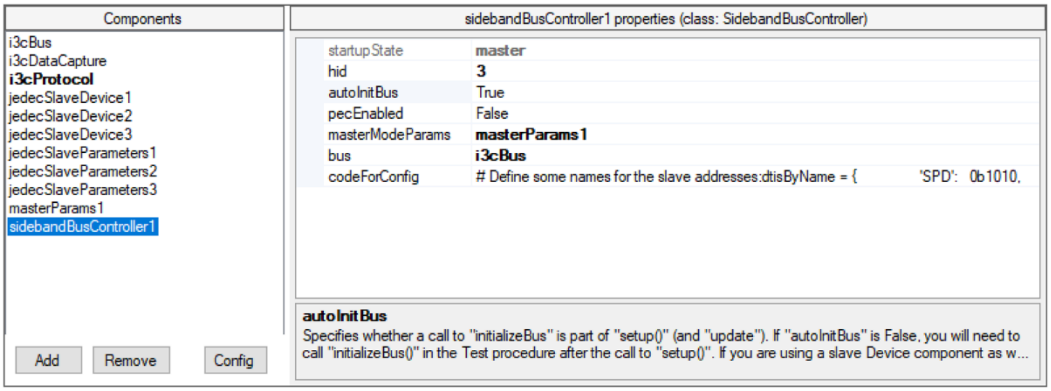


Figure 4: Self-contained setup for the design validation of a SPD Hub device under test.

Figure 5 shows the software interface for declaring a main SidebandBus Host Controller that is attached to the DUT. This is done through a SidebandBusController class, which has attributes that are listed on the right-hand side of the figure. Apart from the expected attributes related to enabling/disabling packet error codes or identifying the host address (through the hid attribute), the SidebandBusController class offers a human-readable representation of common device types such as the SPD Hub. This representation is one of many ease-of-use features that are available in the Introspect ESP Software and that greatly facilitate the development of automated test scripts for any DUT.



Components

i3cBus
i3cDataCapture
i3cProtocol
jedecSlaveDevice1
jedecSlaveDevice2
jedecSlaveDevice3
jedecSlaveParameters1
jedecSlaveParameters2
jedecSlaveParameters3
masterParams1
sidebandBusController1

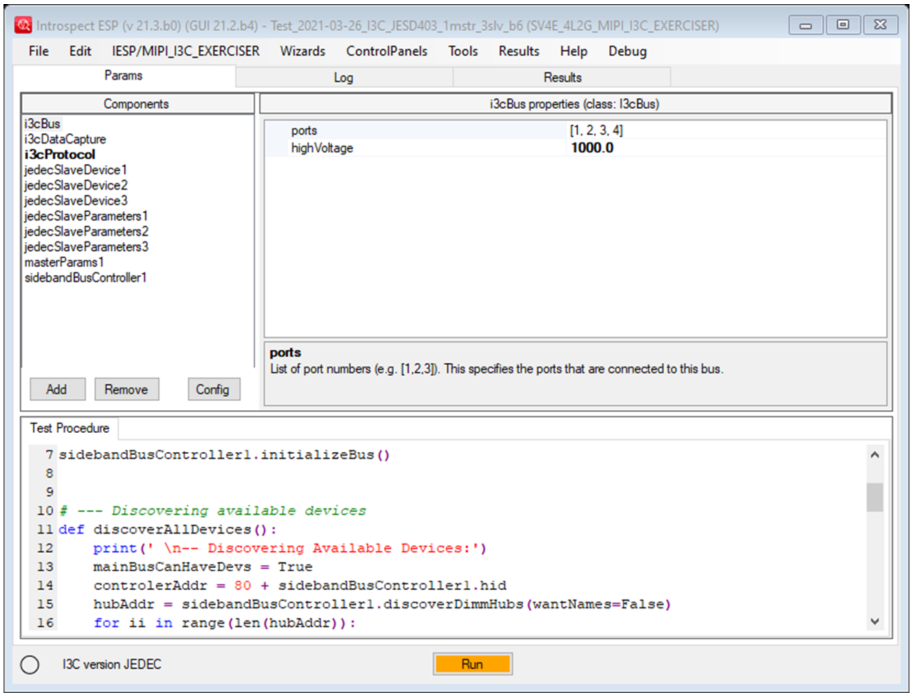
sidebandBusController1 properties (class: SidebandBusController)

startupState	**master**
hid	**3**
autoInitBus	True
pecEnabled	False
masterModeParams	**masterParams1**
bus	**i3cBus**
codeForConfig	# Define some names for the slave addresses:dtisByName = { 'SPD': 0b1010,

autoInitBus
Specifies whether a call to "initializeBus" is part of "setup()" (and "update"). If "autoInitBus" is False, you will need to call "initializeBus()" in the Test procedure after the call to "setup()". If you are using a slave Device component as w...

Figure 5: SidebandBusController component class in the Introspect ESP Software.

Once a SidebandBusController component is declared, it can be used in a scripting environment as shown in Figure 6. In the Test Procedure pane of this figure, there is a call to the method "initializeBus()" which is a high-level method for performing a reset operation and sending broadcast CCCs for operations such as bus configuration (SETBUSCON CCC). Similarly, the component class has a method for automatically discovering hub devices, and this is illustrated in line 15 of the script in Figure 6.


Introspect ESP (v 21.3.b0) (GUI 21.2.b4) - Test_2021-03-26_I3C_JESD403_1mstr_3slv_b6 (SV4E_4L2G_MIPI_I3C_EXERCISER)
File
Edit
IESP/MIPI_I3C_EXERCISER
Wizards
ControlPanels
Tools
Results
Help
Debug
Params
Log
Results
Components
i3cBus properties (class: I3cBus)
Add
Remove
Config
Test Procedure
Run

i3cBus
i3cDataCapture
i3cProtocol
jedecSlaveDevice1
jedecSlaveDevice2
jedecSlaveDevice3
jedecSlaveParameters1
jedecSlaveParameters2
jedecSlaveParameters3
masterParams1
sidebandBusController1

ports	[1, 2, 3, 4]
highVoltage	**1000.0**

ports
List of port numbers (e.g. [1,2,3]). This specifies the ports that are connected to this bus.

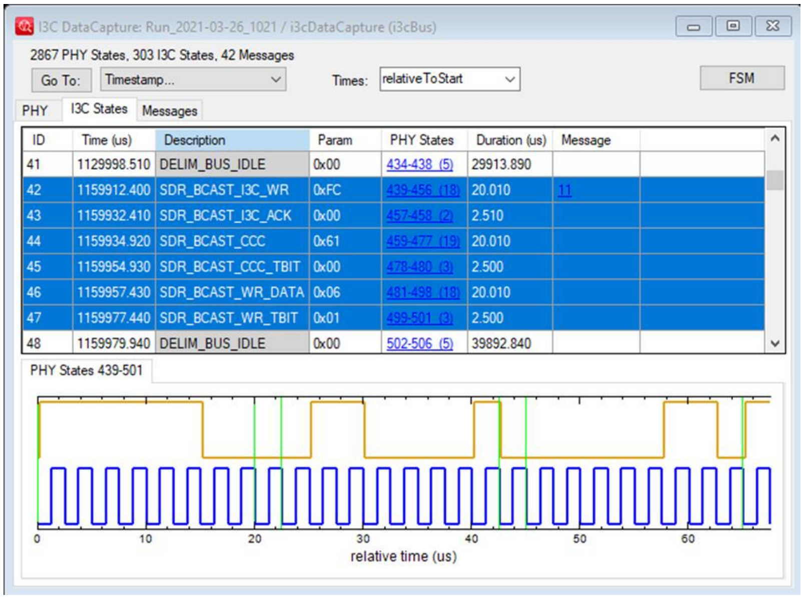
```
 7 sidebandBusController1.initializeBus()
 8
 9
10 # --- Discovering available devices
11 def discoverAllDevices():
12     print(' \n-- Discovering Available Devices:')
13     mainBusCanHaveDevs = True
14     controlerAddr = 80 + sidebandBusController1.hid
15     hubAddr = sidebandBusController1.discoverDimmHubs(wantNames=False)
16     for ii in range(len(hubAddr)):
```

○ I3C version JEDEC

Figure 6: Script example showing bus initialization and SPD Hub discovery.

TESTING ADDRESS MODIFICATION SCHEMES

To test the address modification schemes of the SPD Hub, the SV4E-I3C slave instances can be initialized with the appropriate static address straps for a thermal sensor, PMIC, or RCD. They can then be used to confirm whether the proper address transformation happened within the SPD Hub DUT. Additionally, a protocol analyzer trace can be obtained to verify all states on the SDA and SCL lines of the DUT. An example of a protocol analyzer trace on the output of the DUT is included Figure 7.



I3C DataCapture: Run_2021-03-26_1021 / i3cDataCapture (i3cBus)

2867 PHY States, 303 I3C States, 42 Messages

Go To: Timestamp... Times: relativeToStart

ID	Time (us)	Description	Param	PHY States	Duration (us)	Message
41	1129998.510	DELIM_BUS_IDLE	0x00	434-438 (5)	29913.890	
42	1159912.400	SDR_BCAST_I3C_WR	0xFC	439-456 (18)	20.010	11
43	1159932.410	SDR_BCAST_I3C_ACK	0x00	457-458 (2)	2.510	
44	1159934.920	SDR_BCAST_CCC	0x61	459-477 (19)	20.010	
45	1159954.930	SDR_BCAST_CCC_TBIT	0x00	478-480 (3)	2.500	
46	1159957.430	SDR_BCAST_WR_DATA	0x06	481-498 (18)	20.010	
47	1159977.440	SDR_BCAST_WR_TBIT	0x01	499-501 (3)	2.500	
48	1159979.940	DELIM_BUS_IDLE	0x00	502-506 (5)	39892.840	

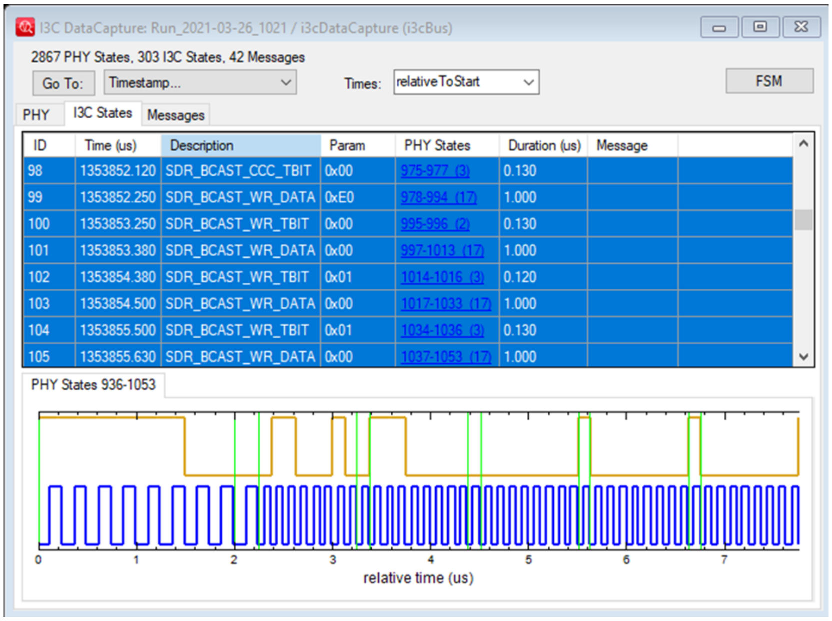
Figure 7: Example trace of the SETHID operation as generated by the master side of the SPD Hub device under test.

PERFORMING READ AND WRITE OPERATIONS

Once address transformation has been confirmed, the SV4E-I3C script can be expanded to perform private write and read operations to/from the SPD Hub and to/from slaves behind the SPD Hub. For operations targeting slaves behind the SPD Hub DUT, the SV4E-I3C ports 2-4 are the ones that will respond to the write/read requests! That is, the master on port 1 of the SV4E-I3C will initiate the command, the command will propagate through the SPD Hub DUT, and then the command will be received by the SV4E-I3C slaves in any of ports 2-4 depending on the address used. This end-to-end capability provides *complete* test coverage for the DUT. Additionally, in cases where a read operation is performed, the slaves on the SV4E-I3C can be pre-programmed with known responses for easy pass/fail testing.

As part of the read/write testing of the SPD Hub, it is important to verify the behavior of packet error checking. With reference to Figure 5, the SidebandBusController class in the Introspect ESP Software provides a convenience method for enabling and disabling PEC transmissions. Additionally, low-level methods are available for inserting errors and verifying CRC calculations.

Figure 8 shows a write operation through a sample SPD Hub DUT. Apart from the protocol payload shown in the table, note the speed change that occurred at around the 2.2 μs mark in the timing diagram at the bottom of the figure. This speed change is important in the SidebandBus protocol, and it is a crucial test item. The frequency of the SV4E-I3C in both the low-speed (I2C) mode and the high-speed mode is programmable, and this enables shmoo testing and other characterization activities.



I3C DataCapture: Run_2021-03-26_1021 / i3cDataCapture (i3cBus)

2867 PHY States, 303 I3C States, 42 Messages

Go To: Timestamp...

Times: relativeToStart

ID	Time (us)	Description	Param	PHY States	Duration (us)	Message
98	1353852.120	SDR_BCAST_CCC_TBIT	0x00	975-977 (3)	0.130	
99	1353852.250	SDR_BCAST_WR_DATA	0xE0	978-994 (17)	1.000	
100	1353853.250	SDR_BCAST_WR_TBIT	0x00	995-996 (2)	0.130	
101	1353853.380	SDR_BCAST_WR_DATA	0x00	997-1013 (17)	1.000	
102	1353854.380	SDR_BCAST_WR_TBIT	0x01	1014-1016 (3)	0.120	
103	1353854.500	SDR_BCAST_WR_DATA	0x00	1017-1033 (17)	1.000	
104	1353855.500	SDR_BCAST_WR_TBIT	0x01	1034-1036 (3)	0.130	
105	1353855.630	SDR_BCAST_WR_DATA	0x00	1037-1053 (17)	1.000	

Figure 8: Example of speed change in the JESD403-1 specification – SidebandBus controller performing a write operation.

Conclusion

This white paper focused on the design and validation of the **Serial Presence Detect EEPROM with Hub Function** (SPD5 Hub or simply SPD Hub), which operates using the SidebandBus Protocol. We introduced the hub architecture in DDR5 and presented some of the design challenges associated with it, such as address modification and packet error codes. We then presented a detailed test methodology for validating and characterizing all ports on an SPD Hub device using the **SV4E-I3C I3C Test and Debug Module** from Introspect Technology, which addresses the main design challenges associated with SPD Hub devices.

Revision Number	History	Date
1.0	Document Release	March 31, 2021

The information in this document is subject to change without notice and should not be construed as a commitment by Introspect Technology. While reasonable precautions have been taken, Introspect Technology assumes no responsibility for any errors that may appear in this document.

Published in Canada on March 31, 2021
EN-W002E-E-21090

INTROSPECT.CA