



Specification for I3CSM

Improved Inter Integrated Circuit

**Version 1.0
23 December 2016**

MIPI Board Adopted 31 December 2016

Public Release Edition

Further technical changes to this document are expected as work continues in the Sensor Working Group.

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

Contents

Contents	iii
Figures.....	vii
Tables.....	x
Release History	xiii
1 Introduction	1
1.1 Scope	2
1.2 I3C Purpose.....	2
1.3 I3C Key Features	3
2 Terminology	5
2.1 Use of Special Terms.....	5
2.2 Definitions.....	5
2.3 Abbreviations	8
2.4 Acronyms.....	8
3 References	10
3.1 Normative References.....	10
3.2 Informative References	10
4 Technical Overview (Informative)	11
4.1 I3C Fundamental Principles.....	12
4.2 I3C Master and Slave Devices	15
4.2.1 I3C Master Device	16
4.2.1.1 I3C Master Device Roles	17
4.2.2 I3C Slave Device	18
4.2.2.1 I3C Slave Device Roles	19
5 I3C Protocol	20
5.1 Single Data Rate (SDR) Mode	20
5.1.1 Bus Configuration	21
5.1.1.1 I3C Device Characteristics.....	21
5.1.1.2 I3C Characteristics Registers	24
5.1.1.2.1 Bus Characteristics Register (BCR).....	25
5.1.1.2.2 Device Characteristics Register (DCR)	26
5.1.1.2.3 Legacy Virtual Register (LVR)	26
5.1.2 Bus Communication	27
5.1.2.1 Role of I3C Slave.....	28
5.1.2.2 I3C Address Header.....	29
5.1.2.2.1 I3C Address Arbitration	30
5.1.2.2.2 I3C Address Arbitration Optimization	30
5.1.2.2.3 Consequence of Master Starting a Frame with an I3C Slave Address.....	31
5.1.2.2.4 Address Header Following a Repeated START is Push-Pull.....	32
5.1.2.2.5 I3C Slave Address Restrictions.....	32
5.1.2.3 I3C SDR Data Words.....	34
5.1.2.3.1 Transition from Address ACK to SDR Master Write Data	34
5.1.2.3.2 Ninth Bit of SDR Master Written Data as Parity	35
5.1.2.3.3 Ninth Bit of SDR Slave Returned (Read) Data as End-of-Data	35

5.1.2.4	Use of Clock Speed to Prevent Legacy I²C Devices From Seeing I3C Traffic	36
5.1.2.4.1	Use of Duty Cycle to Achieve Lower Effective Speed in a Mixed Fast Bus	36
5.1.2.5	Master Clock Stalling	37
5.1.2.5.1	I3C/I ² C Transfer, ACK/NACK Phase	38
5.1.2.5.2	Write Data Transfer, Parity Bit	39
5.1.2.5.3	I3C Read Transfer, Transition Bit	39
5.1.2.5.4	Dynamic Address Assignment, First Bit of Assigned Address	42
5.1.3	Bus Conditions	43
5.1.3.1	Open Drain Pull-Up and High-Keeper	43
5.1.3.2	Bus Free Condition	44
5.1.3.3	Bus Available Condition	44
5.1.3.4	Bus Idle Condition	44
5.1.3.5	Activity States	44
5.1.4	Bus Initialization and Dynamic Address Assignment Mode	45
5.1.4.1	Device Requirements for Dynamic Address Assignment	46
5.1.4.1.1	Unique Identifiability	46
5.1.4.2	Bus Initialization Sequence with Dynamic Address Assignment	46
5.1.4.3	Provisional ID Collision Detection and Correction	49
5.1.5	Hot-Join Mechanism	50
5.1.5.1	Failsafe Device Pads	51
5.1.6	In-Band Interrupt	52
5.1.6.1	Priority Level	52
5.1.6.2	I3C Slave Interrupt Request	52
5.1.6.3	I3C Secondary Master Requests to be Current Master	53
5.1.6.4	I3C Main Master Initiating a Transaction	54
5.1.7	Secondary Master Functions	55
5.1.7.1	Hardware and Software Requirements	55
5.1.7.2	Bus Management Procedures	55
5.1.7.3	Reduced Functionality Secondary Masters	56
5.1.7.4	In-Band Interrupt Handling	56
5.1.7.5	Hot-Join Management	56
5.1.8	Timing Control	57
5.1.8.1	General Principles	57
5.1.8.2	Synchronous Systems and Events	58
5.1.8.3	Asynchronous Systems and Events	61
5.1.8.3.1	Async Mode 0: Asynchronous Basic Mode	66
5.1.8.3.2	Async Mode 1: Asynchronous Advanced Mode	68
5.1.8.3.3	Async Mode 2: Async High-Precision Low-Power Mode	70
5.1.8.3.4	Async Mode 3: Async High-Precision Triggereable Mode	73
5.1.9	Common Command Codes (CCC)	76
5.1.9.1	CCC Command Format	76
5.1.9.2	Broadcast CCCs vs Direct CCCs	77
5.1.9.2.1	End of a CCC Command	77
5.1.9.2.2	Framing Model for Direct CCC Commands	78
5.1.9.2.3	Retry Model for Direct GET CCC Commands	78

5.1.9.3	CCC Command Definitions	80
5.1.9.3.1	Enable/Disable Slave Events Command (ENEC/DISEC)	84
5.1.9.3.2	Enter Activity State 0–3 (ENTAS0–ENTAS3)	85
5.1.9.3.3	Reset Dynamic Address Assignment (RSTDAA).....	85
5.1.9.3.4	Enter Dynamic Address Assignment (ENTDAA).....	86
5.1.9.3.5	Set/Get Max Write Length (SETMWL/GETMWL).....	86
5.1.9.3.6	Set/Get Max Read Length (SETMRL/GETMRL).....	87
5.1.9.3.7	Define List of Slaves (DEFSLVS).....	87
5.1.9.3.8	Enter Test Mode (ENTTM)	88
5.1.9.3.9	Enter HDR Mode 0–7 (ENTHDR0–ENTHDR7)	88
5.1.9.3.10	Set Dynamic Address from Static Address (SETDASA).....	89
5.1.9.3.11	Set New Dynamic Address (SETNEWDA).....	90
5.1.9.3.12	Get Provisional ID (GETPID)	90
5.1.9.3.13	Get Bus Characteristics Register (GETBCR).....	90
5.1.9.3.14	Get Device Characteristics Register (GETDCR).....	90
5.1.9.3.15	Get Device Status (GETSTATUS).....	91
5.1.9.3.16	Get Accept Mastership (GETACCMST)	92
5.1.9.3.17	Set Bridge Targets (SETBRGTGT)	93
5.1.9.3.18	Get Max Data Speed (GETMXDS)	93
5.1.9.3.19	Get HDR Capability (GETHDRCAP).....	95
5.1.9.3.20	Set Exchange Timing Information (SETXTIME).....	95
5.1.9.3.21	Get Exchange Timing Support Information (GETXTIME).....	97
5.1.10	Error Detection and Recovery Methods for SDR.....	98
5.1.10.1	SDR Error Detection and Recovery Methods for I3C Slave Devices	98
5.1.10.1.1	Error Type S0.....	99
5.1.10.1.2	Error Type S1	99
5.1.10.1.3	Error Type S2.....	99
5.1.10.1.4	Error Type S3.....	99
5.1.10.1.5	Error Type S4.....	100
5.1.10.1.6	Error Type S5.....	100
5.1.10.1.7	Error Type S6 (Optional)	100
5.1.10.2	SDR Error Detection and Recovery Methods for I3C Master Devices	100
5.1.10.2.1	Error Type M0	101
5.1.10.2.2	Error Type M1 (Optional).....	101
5.1.10.2.3	Error Type M2	101
5.1.10.2.4	Master Error Detection and Escalation Handling	102
5.2	High Data Rate (HDR) Modes.....	103
5.2.1	HDR Exit Pattern and HDR Restart Pattern	104
5.2.1.1	HDR Exit Pattern.....	104
5.2.1.2	HDR Restart Pattern	105
5.2.1.3	HDR Exit Pattern Detector	105
5.2.1.4	HDR Restart and Exit Pattern Detector	107
5.2.1.5	Compatibility of HDR Pattern Detection and Ternary Modes	109

5.2.2	HDR Double Data Rate Mode (HDR-DDR)	110
5.2.2.1	HDR-DDR Overview	113
5.2.2.2	HDR-DDR Command Coding	117
5.2.2.3	HDR-DDR Bus Turnaround	119
5.2.2.3.1	Command to Read Data from Slave	119
5.2.2.3.2	End of a Read Command Message	120
5.2.2.3.3	Master Termination of a Read Command Message	120
5.2.2.4	HDR-DDR Error Detection.....	121
5.2.2.5	HDR-DDR CRC5 Algorithm.....	122
5.2.3	HDR Ternary Modes (HDR-TSP and HDR-TSL)	123
5.2.3.1	HDR Ternary Signaling and Coding Protocol.....	123
5.2.3.1.1	Ternary Signaling.....	123
5.2.3.1.2	Ternary Coding Protocol.....	124
5.2.3.2	HDR Ternary Command Coding.....	131
5.2.3.3	HDR Ternary Bus Turnaround.....	133
5.2.3.4	HDR Ternary Error Detection.....	134
6	I3C Electrical Specifications	135
6.1	DC I/O Characteristics.....	135
6.2	Timing Specification.....	139
Annex A	I3C Communication Format Details	159
A.1	I3C CCC Transfers.....	159
A.2	I3C Private Write and Read Transfers	160
A.3	Legacy I ² C Write and Read Transfers on the I3C Bus.....	162
A.4	Dynamic Address and Enter HDR	163
Annex B	SDR Mode Error Type Origins	165
B.1	Error Types in I3C CCC Transfers.....	165
B.2	Error Types in I3C Private Read and Write Transfers	166
B.3	Error Types in Dynamic Address Arbitration.....	168
Annex C	I3C Master FSMs	169
Annex D	Typical I3C Protocol Communications	177

Figures

Figure 1 I3C System Diagram.....	1
Figure 2 Energy Consumption and Raw Data Rate: I3C vs. I ² C.....	3
Figure 3 Energy Consumption Comparison for I3C Data Modes.....	4
Figure 4 Example of Data Traffic on the I3C Bus.....	12
Figure 5 I3C Communication Flow.....	13
Figure 6 I3C Master Device Block Diagram.....	16
Figure 7 I3C Slave Device Block Diagram.....	18
Figure 8 I3C Bus with I ² C Devices and I3C Devices.....	21
Figure 9 Address Header Comparison.....	27
Figure 10 Address Arbitration During Header.....	31
Figure 11 Master Clock Stalling in ACK Phase.....	38
Figure 12 Master Clock Stalling in Write Parity Bit.....	39
Figure 13 Master Clock Stalling in T-Bit Before Next Read Data.....	39
Figure 14 Master Clock Stalling in T-Bit Before STOP.....	40
Figure 15 Master Clock Stalling in Low T-Bit Before Repeated Start.....	40
Figure 16 Master Clock Stalling in High T-Bit Before Repeated START.....	41
Figure 17 Master Clock Stalling in High T-Bit Before Repeated START and STOP.....	41
Figure 18 Master Clock Stalling in Dynamic Address First Bit.....	42
Figure 19 Dynamic Address Assignment Transaction.....	48
Figure 20 IBI Sequence with Mandatory Data Byte.....	52
Figure 21 Synchronization of Sampling Moments.....	59
Figure 22 Synchronization Procedure on a General-Purpose I3C Bus.....	60
Figure 23 Example Graph of Async Mode 0 Timestamp Interpolation.....	62
Figure 24 Example of Asynchronous Mode 0 Timestamp Data Transfer.....	66
Figure 25 Example of Asynchronous Mode 1 Timestamp Data Transfer.....	68
Figure 26 Async Mode 1 Timestamp Interpolation.....	69
Figure 27 Example of Asynchronous Mode 2 Timestamp Data Transfer.....	71
Figure 28 Generating SCLK from SCL and SDA Implementation Example.....	71
Figure 29 Asynchronous Mode 2 Implementation Example.....	72
Figure 30 Example of Asynchronous Mode 3 Triggering.....	74
Figure 31 Example of Asynchronous Mode 3 Timestamp Data Transfer.....	75
Figure 32 Asynchronous Mode 3 Implementation Example.....	75
Figure 33 CCC General Frame Format.....	76

Figure 34 Direct CCC Framing Model.....	78
Figure 35 Example Waveform for Error Type S0.....	99
Figure 36 HDR Exit Pattern and Exit Plus STOP	104
Figure 37 HDR Restart with Next Edge.....	105
Figure 38 Example HDR Exit Pattern Detector (Schematic).....	105
Figure 39 Metastable Changes on SCL and SDA Do Not Break the Exit Pattern Detector.....	106
Figure 40 Combined HDR Restart and Exit Pattern Detector (Schematic).....	107
Figure 41 SDA Moves Low to High.....	109
Figure 42 SDA and SCL Changing Metastable	109
Figure 43 SCL Goes Low While SDA Already High.....	110
Figure 44 HDR-DDR Format	111
Figure 45 DDR Preamble Bits State Diagram.....	113
Figure 46 Unused Bit Before Preamble of First HDR-DDR Command	115
Figure 47 Typical HDR-DDR Mode Frame	116
Figure 48 HDR-DDR Command Code After ENTHDR0	118
Figure 49 HDR-DDR Command Code After HDR Repeated Start Pattern	118
Figure 50 Start of HDR-DDR.....	119
Figure 51 Master Allows Normal Read.....	120
Figure 52 Master Terminates Read.....	120
Figure 53 Typical HDR Ternary Mode Frame.....	123
Figure 54 Twelve Symbols Per Data Word	124
Figure 55 SCL and SDA State Transitions and Their Symbolic Coding	127
Figure 56 HDR-TSP and HDR-TSL Waveforms	129
Figure 57 HDR-TSP Command Code After ENTHDR1.....	131
Figure 58 HDR-TSP Command Code After HDR Repeated Start Pattern.....	132
Figure 59 I3C Legacy Mode Timing	143
Figure 60 tDIG_H and tDIG_L.....	144
Figure 61 I3C Data Transfer – ACK by Slave.....	145
Figure 62 I3C Data Transfer – NACK by Slave.....	146
Figure 63 I3C START Condition Timing	147
Figure 64 I3C STOP Condition Timing.....	147
Figure 65 I3C Master Out Timing	148
Figure 66 I3C Slave Out Timing	148
Figure 67 Master SDR Timing	149
Figure 68 Master DDR Timing.....	149

Figure 69 T-Bit When Slave Ends Read and Master Generates STOP	150
Figure 70 T-Bit When Slave Ends Read and Master Generates Repeated START	151
Figure 71 T-Bit When Slave and Master Agree to Continue Read Message	152
Figure 72 T-Bit When Master Ends Read with Repeated START and STOP	153
Figure 73 T-Bit When Master Ends Read via Repeated START and Further Transfer	154
Figure 74 Master to Master Bus Handoff	155
Figure 75 Ternary Protocol Timing	156
Figure 76 I ² C Spike Filter Behavior	157
Figure 77 I3C CCC Transfers	159
Figure 78 I3C Private Write and Read Transfers with 7'h7E Address	160
Figure 79 I3C Private Write and Read Transfers without 7'h7E Address	161
Figure 80 Legacy I ² C Write and Read Transfers in I3C Bus with 7'h7E Address	162
Figure 81 Legacy I ² C Write and Read Transfers in I3C Bus without 7'h7E Address	162
Figure 82 Dynamic Address Allocation ENTDAACCC Bus Modal	163
Figure 83 Enter HDR Mode CCC Bus Modal	163
Figure 84 Error Type Origins: I3C CCC Transfers	165
Figure 85 Error Type Origins: I3C CCC Private Write & Read Transfers with 7'h7E Address ..	166
Figure 86 Error Type Origins: I3C Private Read Transfers without 7'h7E Address	167
Figure 87 Error Type Origins: Dynamic Address Arbitration	168
Figure 88 I3C Main Master FSM	169
Figure 89 Slave Interrupt Request FSM	170
Figure 90 Dynamic Address Assignment FSM	171
Figure 91 Hot-Join FSM	172
Figure 92 Secondary Master Request FSM	173
Figure 93 Master Regaining Bus Ownership FSM	174
Figure 94 HDR Ternary and HDR-DDR Mode FSMs	175
Figure 95 I2C Legacy Master FSM	176
Figure 96 Example Communication Using I3C Coding SDR	177
Figure 97 Example Communication Using I3C Coding SDR with CCC Direct Addressing	178
Figure 98 Example Communication Using I3C Coding SDR with CCC Broadcast	179
Figure 99 Example Communication Using HDR-DDR Protocol	180
Figure 100 Example Communication Using HDR-TSL Protocol	181
Figure 101 Example Communication Using HDR-TSP Protocol	182

Tables

Table 1 Sensor Classes Addressed by I3C.....	2
Table 2 Roles for I3C Compatible Devices.....	15
Table 3 I3C Devices Roles vs Responsibilities.....	22
Table 4 I ² C Features Allowed in I3C Slaves.....	23
Table 5 Legacy I ² C-Only Slave Categories and Characteristics.....	23
Table 6 Bus Characteristics Register (BCR).....	25
Table 7 I3C Device Characteristics Register (DCR).....	26
Table 8 Legacy I ² C Virtual Register (LVR).....	26
Table 9 I3C Slave Address Restrictions.....	33
Table 10 Available Options for Bus Operating Parameters, Per I3C Bus Configuration.....	36
Table 11 Master Clock Stall Times.....	38
Table 12 Activity States.....	44
Table 13 Asynchronous Timing Control Modes.....	65
Table 14 CCC Frame Field Definitions.....	77
Table 15 I3C Common Command Codes.....	80
Table 16 ENEC/DISEC Format 1: Direct.....	84
Table 17 ENEC/DISEC Format 2: Broadcast.....	84
Table 18 Enable Slave Events Command Byte Format.....	84
Table 19 Disable Slave Events Command Byte Format.....	84
Table 20 ENTASx Format 1: Direct.....	85
Table 21 ENTASx Format 2: Broadcast.....	85
Table 22 Enter Activity State CCCs (ENTASx).....	85
Table 23 RSTDAA Format 1: Direct.....	85
Table 24 RSTDAA Format 2: Broadcast.....	85
Table 25 ENTDAAs Format.....	86
Table 26 Direct SETMWL/GETMWL Format.....	86
Table 27 Broadcast SETMWL Format.....	86
Table 28 Direct SETMRL/GETMRL Format.....	87
Table 29 Broadcast SETMRL Format.....	87
Table 30 DEFSLVS Format.....	87
Table 31 ENTTM Format.....	88
Table 32 ENTTM Test Mode Byte Values.....	88
Table 33 Enter HDR Mode CCCs (ENTHDRx).....	88

Table 34 SETDASA Format 1: Primary	89
Table 35 SETDASA Format 2: Point-to-Point	89
Table 36 SETNEWDA Format	90
Table 37 GETPID Format	90
Table 38 GETBCR Format	90
Table 39 GETDCR Format	90
Table 40 GETSTATUS Format	91
Table 41 GETSTATUS MSb-LSb Format	91
Table 42 GETACCMST Format 1: Accepted	92
Table 43 GETACCMST Format 2: Not Accepted	92
Table 44 GETACCMST Format 3: Incorrect Cancel	92
Table 45 SETBRGTGT Format	93
Table 46 GETMXDS Format 1: Without Turnaround	94
Table 47 GETMXDS Format 2: With Turnaround	94
Table 48 maxWr Byte Format	94
Table 49 maxRd Byte Format	94
Table 50 maxRdTurn Format	94
Table 51 GETHDMCAP Format	95
Table 52 GETHDMCAP Byte Fields	95
Table 53 SETXTIME Format 1: Broadcast	95
Table 54 SETXTIME Format 2: Direct	95
Table 55 SETXTIME Defining Byte Values	96
Table 56 GETXTIME Format	97
Table 57 GETXTIME Supported Modes Byte Fields	97
Table 58 GETXTIME State Byte Fields	97
Table 59 SDR Slave Error Types	98
Table 60 SDR Master Error Types	100
Table 61 HDR-DDR Preamble Values	112
Table 62 HDR-DDR Word Format: Command, Data, Reserved	114
Table 63 HDR-DDR Word Formats	115
Table 64 HDR-DDR Command Word Format	117
Table 65 Read and Write Command Spaces for HDR-DDR Mode	117
Table 66 Parity Bits	125
Table 67 Converting 3-bit Binary Triplet Value to Ternary Symbol Pair	125
Table 68 Converting Ternary Symbol Value to SCL and SDA Level Changes	126

Table 69 HDR Ternary Command Word Format.....	131
Table 70 Read and Write Command Spaces for HDR Ternary Modes.....	131
Table 71 I3C I/O Stage Characteristics Common to Push-Pull Mode and Open Drain Mode.....	136
Table 72 Legacy I ² C Device Requirements When Operating on I3C	138
Table 73 I3C Timing Requirements When Communicating With I ² C Legacy Devices.....	140
Table 74 I3C Open Drain Timing Parameters	141
Table 75 I3C Push-Pull Timing Parameters for SDR and HDR-DDR Modes	142
Table 76 I3C Push-Pull Timing Parameters for HDR-TSP and HDR-TSL Modes	143
Table 77 Timing and Drive for Start of New Frame: No Contention on A7	158
Table 78 Timing and Drive for Start of New Frame: With Contention on A7	158
Table 79 Timing and Drive for Continuation of Frame Using Repeated START	158

Release History

Date	Version	Description
2016-12-31	v1.0	Board Adopted release.

This page intentionally left blank.

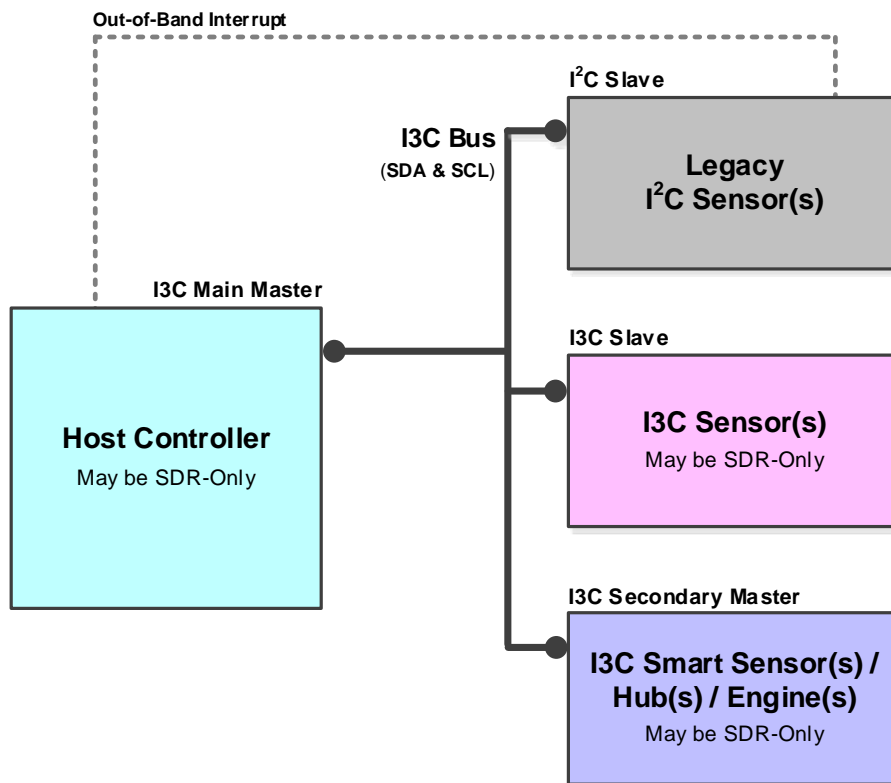
1 Introduction

1 The proliferation of sensors in mobile wireless and mobile-influenced products has created significant design
2 challenges. Because there are no consistent methods for interfacing physical sensors, Device and platform
3 designers are faced with digital interface fragmentation that includes I²C, SPI, and UART among others.

4 In addition to the main interface other signals may be needed, such as dedicated interrupts, chip select signals,
5 and enable and sleep signals. This increases the required number of Host GPIOs, and that in turn drives up
6 system cost with more Host package pins and more PCB layers.

7 As time passes and the number of sensors increases, this situation is becoming increasingly difficult to
8 support and manage.

9 The MIPI I3C interface has been developed to ease sensor system design architectures in mobile wireless
10 products by providing a fast, low cost, low power, two-wire digital interface for sensors.



11
12

Figure 1 I3C System Diagram

13 Example classes of sensor addressed by I3C are listed in *Table 1*.

14

Table 1 Sensor Classes Addressed by I3C

<p>Mechanical / Motion</p> <ul style="list-style-type: none"> · Compass/Magnetometer · Gyro · Accelerometer · Proximity · Touch screen · Grip · Time of Flight (gestures) · Audio/Ultrasonic (events) 	<p>Biometrics/Health</p> <ul style="list-style-type: none"> · Fingerprint · Glucometer · Heart rate · Olfactory (e.g. breathalyzer) · EKG · GSR (galvanic skin response)
<p>Environmental Sensing</p> <ul style="list-style-type: none"> · Ambient light · Barometric pressure / altimeter · Temperature · Carbon monoxide / pollutants · Humidity 	<p>Other</p> <ul style="list-style-type: none"> · NFC (Near Field Communication) · Haptic feedback · IR (smart TV remote) · UV/RGB

1.1 Scope

15 The following topics are in scope for this Specification:

- 16 · I3C interface protocols and commands
- 17 · Electrical specifications, such as timing and voltage levels
- 18 · Support for specific classes of sensors and other Devices

19 The following topics are out of scope for this Specification:

- 20 · Mechanical, system, and implementation details within an I3C Device
- 21 · ESD (Electrostatic Discharge) structures
- 22 · System power management
- 23 · Use case specific data or format definitions besides Bus management command codes

1.2 I3C Purpose

24 The I3C interface is intended to improve upon the features of the I²C interface [*NXP01*], preserving backward
 25 compatibility. This Specification defines a standard Multi-Drop interface between Host processors and
 26 peripheral sensors.

27 Implementing the I3C Specification greatly increases the flexibility mobile terminal system designers have
 28 to support an ever-expanding sensor subsystem as efficiently and at as low a cost as possible.

1.3 I3C Key Features

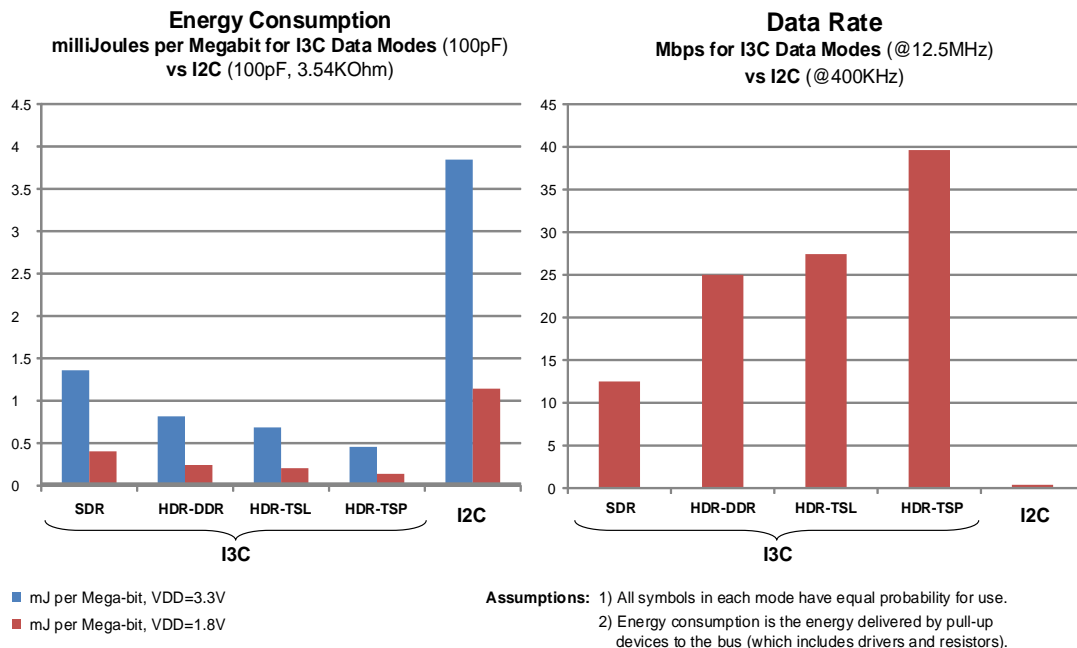
29 Two main concerns are paramount for the I3C interface: The use of as little energy as possible in transporting
30 data and control, while reducing the number of physical pins used by the interface.

31 Therefore, the I3C interface features:

- 32 · Two wire serial interface up to 12.5 MHz using Push-Pull
- 33 · Legacy I²C Device co-existence on the same Bus (with some limitations)
- 34 · Dynamic Addressing while supporting Static Addressing for Legacy I²C Devices
- 35 · Legacy I²C messaging
- 36 · I²C-like Single Data Rate messaging (SDR)
- 37 · Optional High Data Rate messaging Modes (HDR)
- 38 · Multi-Drop capability
- 39 · Multi-Master capability
- 40 · In-Band Interrupt support
- 41 · Hot-Join support
- 42 · Synchronous Timing Support and Asynchronous Time Stamping

43 The I3C interface provides major efficiencies in Bus power while providing greater than 10x speed
44 improvements over I²C. *Figure 2* and *Figure 3* show different Bus Mode options that provide tradeoffs for
45 performance/power vs. target Device complexity.

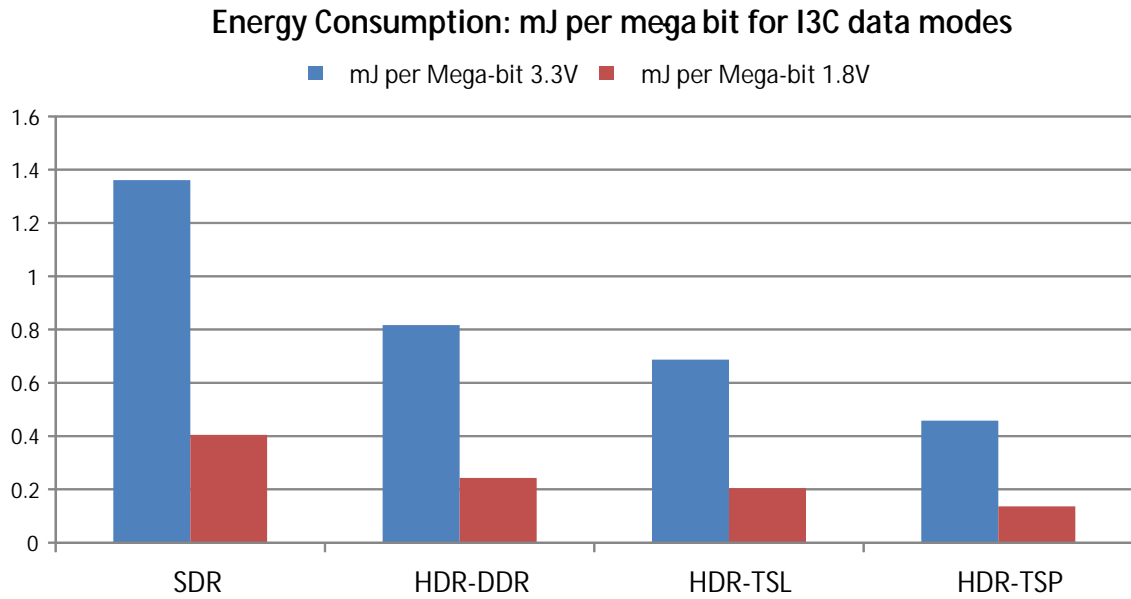
The bar chart on the left shows energy consumption for a given amount of data for the different I3C modes, compared to I2C. The bar chart on the right shows the same comparison for data throughput. Both charts show a significant advantage for I3C.



46 **Figure 2 Energy Consumption and Raw Data Rate: I3C vs. I²C**

48 The bar chart in *Figure 2* shows energy consumption for a given amount of data for the different I3C Modes
49 compared to I²C (units are milli-Joules per megabit) whereas on the right is data-throughput. Both show a
50 significant advantage for I3C.

51 **Figure 3** shows an expanded view of energy consumption for different I3C Modes.



Assumptions: 1) All symbols in each mode have equal probability for use.
 2) Energy consumption is the energy delivered by-pull up devices to the bus (which includes drivers and resist

52

53

54

Figure 3 Energy Consumption Comparison for I3C Data Modes

2 Terminology

2.1 Use of Special Terms

55 The MIPI Alliance has adopted Section 13.1 of the *IEEE Standards Style Manual*, which dictates use of the
56 words “shall”, “should”, “may”, and “can” in the development of documentation, as follows:

57 The word *shall* is used to indicate mandatory requirements strictly to be followed in order
58 to conform to the Specification and from which no deviation is permitted (*shall equals is*
59 *required to*).

60 The use of the word *must* is deprecated and shall not be used when stating mandatory
61 requirements; *must* is used only to describe unavoidable situations.

62 The use of the word *will* is deprecated and shall not be used when stating mandatory
63 requirements; *will* is only used in statements of fact.

64 The word *should* is used to indicate that among several possibilities one is recommended
65 as particularly suitable, without mentioning or excluding others; or that a certain course of
66 action is preferred but not necessarily required; or that (in the negative form) a certain
67 course of action is deprecated but not prohibited (*should equals is recommended that*).

68 The word *may* is used to indicate a course of action permissible within the limits of the
69 Specification (*may equals is permitted to*).

70 The word *can* is used for statements of possibility and capability, whether material,
71 physical, or causal (*can equals is able to*).

72 All sections are normative, unless they are explicitly indicated to be informative.

73 All quoted voltage and frequency values in the informative sections represent their typical values.

2.2 Definitions

74 **ACK:** Short for “acknowledge”. See also **NACK**.

75 **Address Arbitration:** Process for determining arbitrated Addresses to resolve contention.

76 **Address:** A set of bits designating a Device or the location of a register.

77 **Arbitrable:** Subject to decision by Arbitration.

78 **Arbitration:** If two Devices start transmission at the same time, then Arbitration is required to determine
79 Bus control. Arbitration could also be required during a Slave transmission if a Master addresses multiple
80 Slaves.

81 **Bridge Device:** Device on the I3C Bus that allows conversion from the native I3C Bus protocol to another
82 protocol (such as SPI, UART etc.).

83 **Broadcast:** A command intended for multiple Slave Devices, using the Broadcast Address 7h7E.

84 **Bus:** The physical and logical implementation of the SCL and SDA lines.

85 **Bus Available Condition:** State on the I3C Bus where both the SCL line and the SDA line are High for at
86 least t_{AVAIL} (see **Table 74**), and a Device is able to initiate a transaction on the Bus.

87 **Bus Free Condition:** State on the I3C Bus after a STOP and before a START for at least t_{CAS} (see **Table 74**).

88 **Bus Idle Condition:** An extended duration of the Bus Free Condition that indicates that Devices may attempt
89 to Hot-Join the I3C Bus.

90 **Bus Turnaround:** When a transmitting Device sends a command, and then the receiving Device takes over
91 the I3C Bus in order to respond.

92 **Characteristics:** Quantification of a Device’s available features and capabilities.

- 93 **Clock to Data Turnaround Time:** The time duration between reception of an SCL edge and the start of
94 driving an SDA change. See t_{SCO} in *Table 75*.
- 95 **CRC5:** Cyclic Redundancy Check with fifth-order polynomial length.
- 96 **Current Master:** The I3C Device that presently has Master control of the I3C Bus.
- 97 **Device:** A Master or Slave.
- 98 **Device ID:** Defines a Device's characteristic or function within a sensor system.
- 99 **Dynamic Address:** A Device Address that is assigned or allocated during initialization of the Bus. Usually
100 occurs after power up.
- 101 **Failsafe:** An I3C Device Bus pad is considered Failsafe if its leakage does not increase when it is unpowered.
102 A pad may be unpowered because the Device is unpowered, because its IO rail is unpowered or clamped, or
103 both. Failsafe only matters for some Hot-Join Devices.
- 104 **Frame:** A Frame begins with a START, followed by the Address of the targeted Slave(s), Data, and finally a
105 STOP.
- 106 **High:** Defines a signal level that is a logical "1".
- 107 **High Data Rate (HDR):** High Data Rate Modes that achieve higher speed by transferring data on both clock
108 edges.
- 109 **High-Keeper:** A weak Pull-Up type Device used when SDA, and sometimes SCL, is in High-Z with respect
110 to all Devices.
- 111 **Host:** Hardware and software that provides the core functionality of a mobile device.
- 112 **Hot-Join:** Slaves that join the Bus after it is already started, whether because they were not powered
113 previously or because they were physically inserted into the Bus; the Hot-Join mechanism allows the Slave
114 to notify the Master that it is ready to get a Dynamic Address.
- 115 **In-Band Interrupt (IBI):** A method whereby a Slave Device emits its Address into the arbitrated Address
116 header on the I3C Bus to notify the Master of an interrupt.
- 117 **I²C Device:** A Master or Slave that meets the requirements of the I²C Specification [*NXP01*].
- 118 **I3C Device:** A Master or Slave that meets the requirements of the I3C Specification.
- 119 **I3C Slave:** See Slave.
- 120 **Legacy I²C:** I3C maintains the industry standard architecture of I²C and supports existing I²C Slave Devices.
121 I3C does not support I²C Bus Masters.
- 122 **Low:** Defines a signal level that is a logical "0".
- 123 **Main Master:** Master that has overall control of the I3C Bus. Including control and hand off to Secondary
124 Masters.
- 125 **Master:** A reference to the I3C Bus Device that is controlling the Bus.
- 126 **Mastership:** Control of the I3C Bus, in a Master role.
- 127 **Message:** A packetized communication between Devices.
- 128 **Minimal Bus:** An I3C Bus with one Master Device (potentially with reduced functionality), and one active
129 Slave Device with a fixed and reserved Slave Address value of 7'h01. Additional Read-only Slave Devices
130 may optionally also be present in a Minimal Bus, but there can be no additional Read-Write Slave Devices.
- 131 **MIPI Manufacturer ID:** A two byte/16 bit unique identifier for a vendor of a MIPI compliant Device
132 [*MIPI01*].
- 133 **Mixed Fast Bus:** I3C Bus topology with both I²C and I3C Devices present on the I3C Bus, where the I²C
134 Devices have a true I²C 50 ns Spike Filter on the SCL line.

- 135 **Mixed Slow/Limited Bus:** I3C Bus topology with both I²C and I3C Devices present on the I3C Bus, where
136 the I²C Devices do not have a true I²C 50 ns Spike Filter on the SCL line.
- 137 **Mode:** Distinguishes different data transfer methods used in I3C including Legacy I²C Mode, Single Data
138 Rate Mode (SDR), High Data Rate Mode (HDR), Dual Data Rate Mode (HDR-DDR), Ternary Symbol
139 Legacy Mode (HDR-TSL) and Ternary Symbol for Pure Bus Mode (HDR-TSP).
- 140 **Multi-Drop:** A Bus that communicates through a process of Arbitration to determine which Device sends
141 information at any point. The other Devices listen for data they are intended to receive.
- 142 **Multi-Master:** Multiple Bus Masters present on the Bus. Used when multiple nodes on the Bus need to
143 initiate a transfer.
- 144 **NACK:** Short for “not acknowledge”, which means No ACK was asserted. See also **ACK**.
- 145 **Offline Capable:** An Offline Capable Device is able to disconnect from the physical I3C Bus and/or is able
146 to ignore I3C traffic on the I3C Bus. A Device’s Offline capability is one of the capabilities reflected in its
147 Bus Characterization Register.
- 148 **Open Drain:** High-Z with an active Pull-Down. Typically used in conjunction with a passive Pull-Up.
- 149 **Park:** Logic level High set by the Master (or Slave on Read) before turning around the Bus to allow the other
150 to drive to Logic level Low or not (will be held High by weak Pull-Up).
- 151 **Preamble:** The bits preceding the data words in HDR-DDR.
- 152 **Pull-Down:** Active mechanism used to pull the Bus to a logical Low state.
- 153 **Pull-Up:** Mechanism used to pull the Bus to a logical High state. The mechanism may be either active or
154 passive.
- 155 **Pure Bus:** A Bus topology with only I3C Devices present. No I²C Devices are permitted on a Pure Bus.
- 156 **Push-Pull:** Active Pull-Down and active Pull-Up on output driver.
- 157 **Repeated START:** Two or more instances of a START in a row without an intervening STOP. A Repeated
158 START is used in circumstances where the Master wishes to continue communicating on the I3C Bus without
159 having to first generate a STOP. In this Specification, a Repeated START is abbreviated as “Sr”. This is
160 equivalent to repeated START in I²C [*NXP01*].
- 161 **SDR-Only:** An SDR-Only Device supports only SDR Mode, i.e. does not support HDR Mode.
- 162 **Secondary Master:** A Secondary Master controls the I3C Bus only after receiving permission from the Main
163 Master. Control of the Bus is temporary, and always ends with passing control back to the Main Master.
- 164 **Single Data Rate (SDR):** Single Data Rate transfers data on only one edge of the clock.
- 165 **Slave:** A Slave Device can only respond to either Common or individual commands from a Master. A Slave
166 Device cannot generate a clock.
- 167 **Spike Filter:** A filter that removes SCL (and SDA) spikes shorter than 50 ns in duration. See Input Filter (t_{SP}
168 parameter) in the I²C Specification [*NXP01*]. Also known as a glitch filter.
- 169 **Stall:** The act of the I3C Master holding the SCL line Low under specific transitory conditions.
- 170 **START:** START is the I3C Bus condition of a High to Low transition on the SDA line while the SCL line
171 remains High. In this Specification, a START is abbreviated as “S”.
- 172 **START Request:** A method for a Slave to force the Master to issue a START on an idled I3C Bus.
- 173 **Static Address:** A Device Address that is fixed and cannot be changed.
- 174 **STOP:** STOP is the I3C Bus condition of a Low to High transition on the SDA line while the SCL line
175 remains High. In this Specification, a STOP is abbreviated as “P”.
- 176 **Symbol:** In I3C Ternary Coding, a Symbol is an abstraction that can have any of three states. These states
177 represent whether (for a given period on the I3C Bus) the SCL line changes state, the SDA line changes state,
178 or both lines change state. If neither line changes state, then no Symbol is generated for that period.

- 179 **T-Bit:** Transition bit, an alternative to the ACK/NACK mechanism.
- 180 **Ternary Mode:** I3C HDR-TSP (Ternary Symbol for Pure Bus) Mode or HDR-TSL (Ternary Symbol Legacy)
- 181 Mode.
- 182 **Timestamping:** The act of determining and assigning the time at which an event occurred.
- 183 **Timing Control:** Methods of exchanging and controlling system timing information, with the intent to
- 184 synchronize and/or Timestamp I3C Bus Devices and events. See *Section 5.1.8*.
- 185 **Word:** Transmission containing 16 payload bits and two parity bits.
- 186 **Word Types:** Four Word Types are used: Command Word, User Data Word, CRC Word, and Reserved Word.

2.3 Abbreviations

- 187 ACK Acknowledge
- 188 e.g. For example (Latin: *exempli gratia*)
- 189 i.e. That is (Latin: *id est*)
- 190 High-Z An output driver that is set to high impedance mode (cannot source or sink current)
- 191 NACK Not Acknowledge
- 192 P STOP
- 193 PHY Physical Layer
- 194 S START
- 195 Sr Repeated START
- 196 T Transition Bit

2.4 Acronyms

- 197 BCR Bus Characteristics Register
- 198 BER Bit Error Rate
- 199 BMB Bus Management Block
- 200 CCC Common Command Code
- 201 CRC Cyclic Redundancy Check
- 202 DAR Dynamic Address Request
- 203 DCR Device Characteristics Register
- 204 DDR Double Data Rate
- 205 ESD Electro Static Discharge
- 206 FSM Finite State Machine
- 207 HDR High Data Rate
- 208 HDR-DDR HDR Double Data Rate Mode
- 209 IBI In-Band Interrupt
- 210 ISTO Industry Standards and Technology Organization
- 211 LCR Legacy Characteristics Register

212	LSb	Least Significant Bit
213	Mbps	Megabits per second
214	MHz	Mega Hertz
215	MID	MIPI Manufacturer Identification [<i>MIPI01</i>]
216	MSb	Most Significant Bit
217	NVMEM	Non-Volatile Memory
218	OD	Open Drain
219	PICS	Protocol Implementation Conformance Statement
220	PUR	Pull-Up Resistor
221	SCL	Serial Clock
222	SDA	Serial Data
223	SDR	Single Data Rate
224	SPI	Serial Peripheral Interface
225	SWG	Sensors Interface Working Group, part of MIPI Alliance
226	TSL	Ternary Symbol Legacy
227	TSP	Ternary Symbol for Pure Bus (no I ² C Devices)
228	UART	Universal Asynchronous Receiver Transmitter

3 References

3.1 Normative References

- 229 [MIPI01] MIPI Alliance, Inc., “MIPI Alliance Manufacturer ID Page”, <http://mid.mipi.org>, July 24,
230 2015.

3.2 Informative References

- 231 [NXP01] UM10204, *I²C Bus Specification and User Manual*, Rev. 6 – 4, NXP Corporation, April
232 2014
- 233 [USB01] *Universal Serial Bus 3.1 Specification*, Revision 1.0,
234 http://www.usb.org/developers/docs/usb_31_072715.zip, Hewlett-Packard Company *et*
235 *al.*, July 26, 2013

4 Technical Overview (Informative)

236 This Section generally describes the I3C Bus, the I3C interface, and I3C Master and Slave Devices.
237 I3C is a two-wire bidirectional serial Bus, optimized for multiple sensor Slave Devices and controlled by
238 only one I3C Master Device at a time. I3C is backward compatible with many Legacy I²C Devices, but I3C
239 Devices also support significantly higher speeds, new communication Modes, and new Device roles,
240 including an ability to change Device Roles over time (i.e., the initial Master can cooperatively pass the
241 Master role to another I3C Device on the Bus, if the second I3C Device supports that feature).
242 I3C includes:
243 • Support for many Legacy I²C Slave Devices and messages
244 • **I3C Single Data Rate (SDR) Mode:** New I3C enhanced version of the I²C protocol supporting
245 private messages, and adding two kinds of standard built-in messages:
246 • **Broadcast messages**, which are sent to all I3C Slaves on the Bus
247 • **Direct messages**, which are addressed to specific Slaves
248 • **I3C High Data Rate (HDR) Modes:** Additional optional Modes that add significant
249 functionality:
250 • **Dual Data Rate (HDR-DDR) Mode:** Uses same signaling as SDR Mode (i.e. is not
251 significantly different from the I²C protocol), but runs at about 2x the speed of SDR
252 • **Ternary Symbol Legacy (HDR-TSL) Mode:** Higher data rates plus Ternary coding, for Buses
253 with a mix of I2C and I3C Devices. Significantly different from the I²C protocol
254 • **Ternary Symbol Pure-bus (HDR-TSP) Mode:** Higher data rates plus Ternary coding, for
255 Buses with only I3C Devices. Significantly different from the I²C protocol
256 Readers interested in a detailed, low-level introduction to the operation of the I3C Bus for each of the defined
257 I3C Modes are encouraged to study and compare the example waveforms shown in informative *Annex D*.

4.1 I3C Fundamental Principles

258 I3C supports several communication formats, all sharing a two-wire interface.

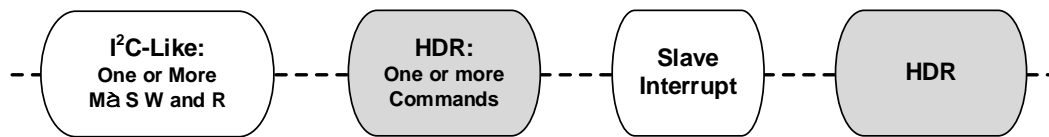
259 The two wires are designated SDA and SCL:

- 260 · SDA (Serial Data) is a bidirectional data pin
- 261 · SCL (Serial Clock) can be either a clock pin or a Bi-directional data pin while in certain HDR
- 262 Modes

263 An I3C Bus supports the mixing of various Message types:

- 264 1. I²C-like SDR Messages, with SCL clock speeds up to 12.5MHz
- 265 2. Broadcast and Direct Common Command Code (CCC) Messages that allow the Master to
- 266 communicate to all or one of the Slaves on the I3C Bus, respectively
- 267 3. HDR Mode Messages, which achieve higher data rates per equivalent clock cycle
- 268 4. I²C Messages to Legacy I²C Slaves
- 269 5. Slave-initiated START Request to the Master, for example to send an In-Band Interrupt or to
- 270 request the Master role

271 An example traffic pattern on the I3C Bus is shown in *Figure 4*.

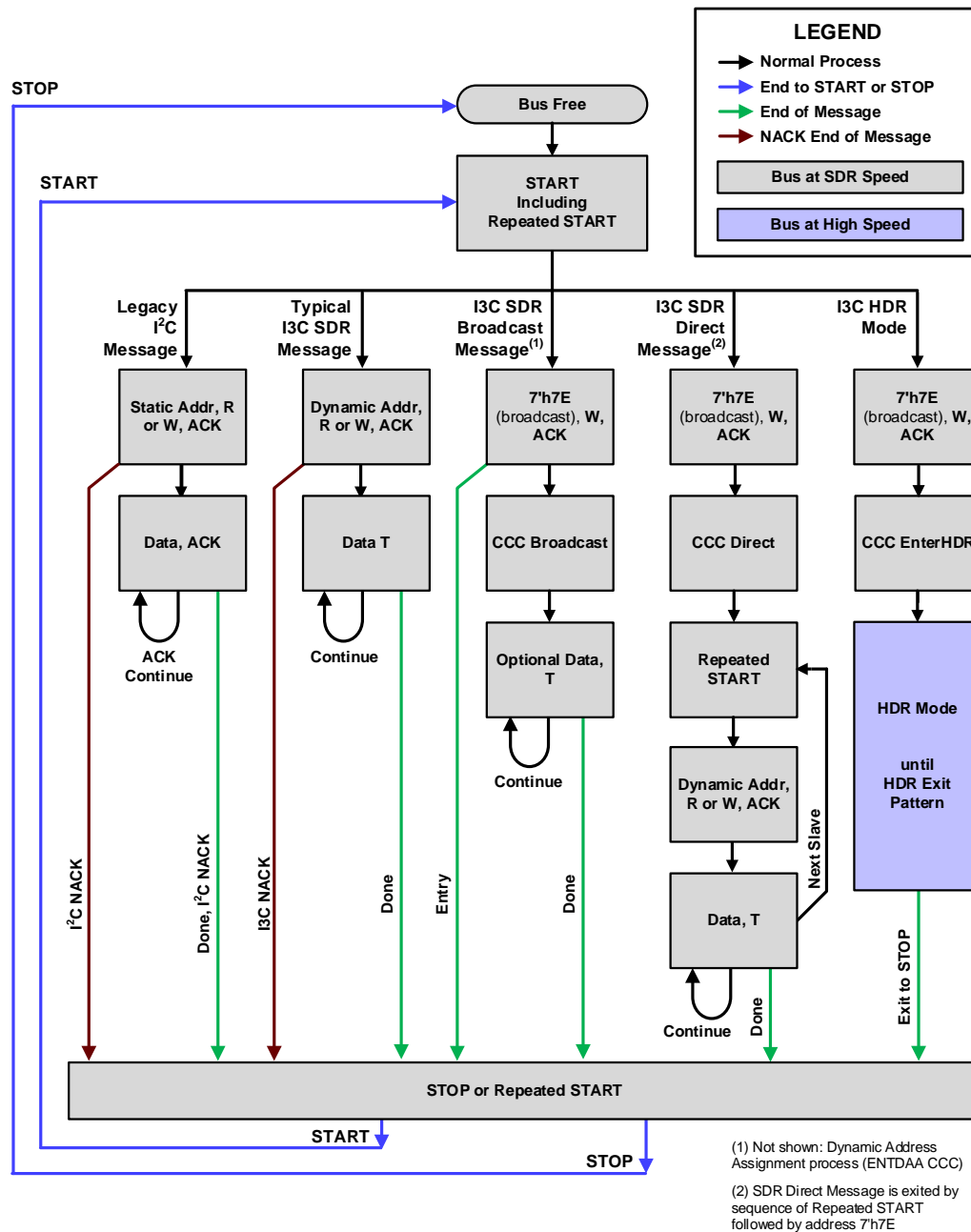


272

273

Figure 4 Example of Data Traffic on the I3C Bus

- 274 **Figure 5** illustrates how I3C communication is initiated:
- 275 • All I3C communication occurs within a Frame. The Frame begins with a START, followed by one
- 276 or more transfers, and a STOP.
- 277 • For the HDR Modes:
- 278 • First the dedicated Broadcast I3C Address (7'h7E) is issued to all Slaves on the I3C Bus.
- 279 • Then one of the EnterHDR CCCs is issued, indicating that the Master is entering an HDR
- 280 Mode. Each HDR Mode has its own EnterHDR CCC.
- 281 • This is followed by one or more HDR transfers.
- 282 • HDR Mode is ended by using the HDR Exit Pattern protocol.



283
284

Figure 5 I3C Communication Flow

285 I3C is based on a Frame encapsulation approach. A Frame includes a Data Payload. The transfer protocol for
286 the Data Payload is either SDR or HDR. One example of HDR is the ternary number symbol system specified
287 in *Section 5.2*. Frames are bordered by I²C-like Bus management.

288 The I3C Frame always includes at least the START, the Header, the Data, and the STOP.

289 The Header following a START allows for Bus Arbitration. The Master uses the Header to address Slave
290 Device(s). Slave Device(s) may use the Header Arbitration for multiple purposes: for In-Band Interrupt, for
291 Hot-Join, and for Secondary Master functionality.

292 Common Command Codes (CCCs) are used to enter the High Data Rate (HDR) Modes. It is important to
293 understand that I3C Bus activity for the HDR Message does not follow the Legacy I²C format. For example,
294 the HDR-TSP payload is encoded using a ternary number symbol system where both the SDA line and the
295 SCL line transfer data, with the clock embedded within the data transfer protocol.

296 I3C allows only one Master to have control of the I3C Bus at a time. Mechanisms for handoff of the Master
297 role from one Device to another Device are provided.

4.2 I3C Master and Slave Devices

298 A given I3C Bus always has one Master and one or more Slaves. This Section generally describes I3C Master
299 Devices and I3C Slave Devices.

300 A given I3C Device can be designed to function either solely as an I3C Master, solely as an I3C Slave, or
301 with both I3C Master and I3C Slave capabilities.

302 An I3C Device with both I3C Master and I3C Slave capabilities cannot function as both Master and Slave at
303 the same time, instead it must be configured either as an I3C Slave Device or as an I3C Master Device. Such
304 an I3C Device can be initially configured (initialized) on an I3C Bus either as the Master of that I3C Bus, or
305 as a Slave on that I3C Bus. However for that I3C Bus to function properly, only one of the multiple I3C
306 Devices on the Bus can be initially configured (initialized) as an I3C Master Device. That I3C Device will
307 have the ‘Main Master’ Device Role, and will be the first I3C Device on the Bus to serve as Current Master;
308 all other I3C Devices and Legacy I²C Devices on the I3C Bus will be initially configured (initialized) as
309 Slaves.

310 I3C introduces the concept of Current Master, defined as the I3C Master Device on the I3C Bus that is
311 functioning as Master (i.e., the one that is controlling the Bus) at the present time. Only one I3C Device on
312 an I3C Bus can serve as Current Master at a time. However after initial Bus configuration the Current Master
313 function can be cooperatively passed from the Current Master to any other I3C Device on the Bus with I3C
314 Master Device capability, using provided I3C commands (CCCs).

315 I3C defines several Master and Slave Device Roles (see *Table 2* and *Table 3*) to reflect the functional
316 capabilities of a given I3C Master or Slave Device. A given I3C Device must support at least one Device
317 Role, and can be designed to support multiple Device Roles. Every I3C Device exposes the Device Roles it
318 supports via its Bus Characteristics Register (BCR, see *Section 5.1.1.2.1*).

319

Table 2 Roles for I3C Compatible Devices

Device Type	Device Role	Description
I3C Master ¹	I3C Main Master	Initially configures I3C Bus, has HDR support
	SDR-Only Main Master	Initially configures I3C Bus, no HDR support
	I3C Secondary Master	Can Master but currently functioning as Slave
	SDR-Only Secondary Master	Can Master but currently functioning as Slave, no HDR support
I3C Slave ²	I3C Slave	Ordinary I3C Slave, no Master capability
	I ² C Slave	No I3C Master or I3C Slave capabilities
Note:		
1) Applies to Master-only Devices. In a Multi-Master context a Master Device may also implement functionality to join the Bus acting in a Slave role.		
2) Applies to Slave-only Devices. In a Multi-Master context a Slave Device may also implement functionality to join the Bus acting in a Master role.		

4.2.1 I3C Master Device

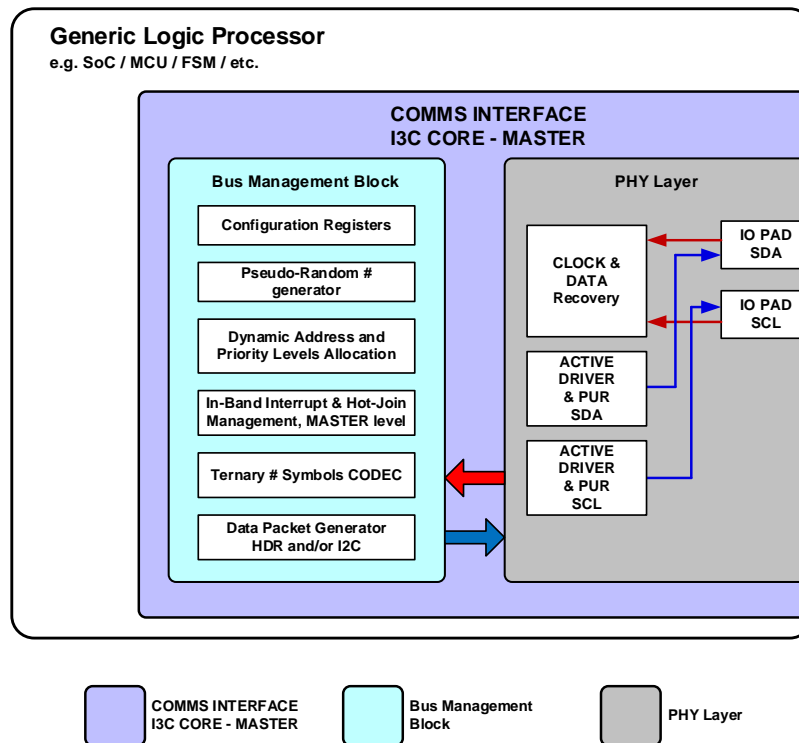
320 An I3C Bus requires there to be exactly one I3C Device at a time functioning as an I3C Master Device. In
 321 I3C terms, this I3C Master Device is the Current Master at that time. In typical applications, the Current
 322 Master is the I3C Device on the Bus that sends the majority of the I3C Commands (CCC), addressing either
 323 all Slaves (Broadcast CCCs) or specific individual Slaves (Directed CCCs). The Current Master is also the
 324 only Device on the I3C Bus allowed to send I²C Messages.

325 In addition to sending I3C Commands and I²C Messages, an I3C Master Device also:

- 326 · Generates the Bus clock when in SDR Mode and DDR Mode (there is no traditional clock in the
- 327 Ternary Modes)
- 328 · Manages Pull-Up structures
- 329 · Manages the Dynamic Address Assignment procedure (including Hot-Join events) while acting as
- 330 the Main Master
- 331 · Manages START Requests from I3C Slave Devices on the Bus along with Address Arbitration
- 332 requests:
- 333 · Generate In-Band Interrupts
- 334 · For Hot-Join events
- 335 · To become Current Master
- 336 · Supports I²C Legacy Slave Devices
- 337 · Supports I3C SDR Mode

338 In addition, an I3C Master Device can optionally support any combination of I3C’s defined HDR Modes.

339 **Figure 6** is a block diagram of a typical generic I3C Master Device.



340
341

Figure 6 I3C Master Device Block Diagram

4.2.1.1 I3C Master Device Roles

342 All I3C Master Devices support one of the two Main Master Device Roles, and may also support one of the
343 two Secondary Master Device Roles.

344 Main Master Device Roles:

- 345 · **Main Master:** The I3C Master Device on the I3C Bus that initially configures the I3C Bus and
346 serves as the first Current Master. Only one I3C Device on a given I3C Bus can take the Main
347 Master role, i.e. the role cannot be passed on to any other I3C Device on the I3C Bus. Supports
348 both SDR Mode and HDR Mode.
- 349 · **SDR-Only Main Master:** A Main Master that only supports I3C's SDR Mode , i.e. does not
350 support any of the HDR Modes.

351 Secondary Master Device Roles:

- 352 · **I3C Secondary Master:** Any I3C Device on the I3C Bus, other than the Current Master, with I3C
353 Master Capability. There can be multiple Secondary Masters on an I3C Bus at the same time. By
354 definition, a Secondary Master functions as an I3C Slave Device until and unless it eventually
355 becomes Current Master. Supports both SDR Mode and HDR Mode.
- 356 · **SDR-Only Secondary Master:** A Secondary Master that only supports I3C's SDR Mode, i.e.
357 does not support any of the HDR Modes.

358 See also *Table 2* and *Table 3*.

359 Note:

360 *Current Master is not formally defined as an I3C Device Role, and is not exposed in the I3C Device's*
361 *Bus Characteristics Register (BCR, see Section 5.1.1.2.1).*

4.2.2 I3C Slave Device

362 An I3C Bus supports up to 11 I3C Slave Devices, though the maximum number of Devices will depend on
 363 trace length, capacitive load per Device, and the types of Devices (I²C vs. I3C) present on the Bus, because
 364 these factors affect clock frequency requirements.

365 An I3C Slave Device listens to I3C Bus for relevant I3C Commands (CCCs) sent by the Current Master, and
 366 responds accordingly. This includes all Broadcast Commands (CCC), and any Directed Commands (CCC)
 367 addressed specifically to that I3C Slave Device and supported by that I3C Slave Device.

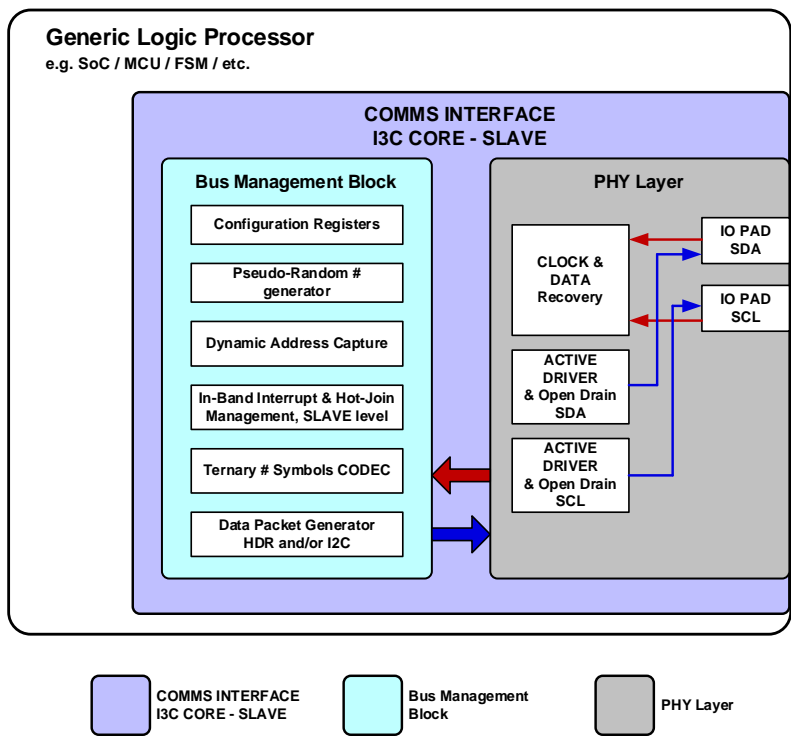
368 In addition to responding to I3C Commands, an I3C Slave Device always supports I3C SDR Mode.

369 In addition, an I3C Slave Device can optionally:

- 370 · Request In-Band Interrupts
- 371 · Generate Hot-Join events
- 372 · Request to become Current Master, if the I3C Slave Device also has I3C Master Device capability
- 373 · Support any combination of I3C's defined HDR Modes

374 While functioning as a Slave, an I3C Slave Device functions in one of the Slave Device Roles detailed in
 375 *Section 4.2.2.1*.

376 *Figure 7* is a block diagram of a typical generic I3C Slave Device.



377
378

Figure 7 I3C Slave Device Block Diagram

4.2.2.1 I3C Slave Device Roles

379 All I3C Slave Devices support one of the two I3C Slave Device Roles:

- 380 · **I3C Slave:** An ordinary I3C Slave Device without Master capability. Supports both SDR Mode
381 and HDR Mode.
- 382 · **SDR-Only I3C Slave:** An I3C Slave without Master capability that only supports I3C's SDR
383 Mode (i.e., does not support any of the HDR Modes).

384 **Note:**

385 *An additional Slave Device Role is defined for $\dot{P}C$ Slave, however this is not relevant for an I3C Slave*
386 *Device.*

387 See also *Table 2* and *Table 3*.

5 I3C Protocol

388 This Section specifies the communication protocols for all defined I3C Modes:

- 389 · **Single Data Rate (SDR) Mode:** See *Section 5.1*
- 390 · **High Data Rate (HDR) Modes:** See *Section 5.2*
- 391 · **HDR Ternary Symbol Pure-bus (HDR-TSP) Mode**
- 392 · **HDR Ternary Symbol Legacy-inclusive-bus (HDR-TSL) Mode**
- 393 · **HDR Double Data Rate (HDR-DDR) Mode**

394 It is important to note that the I3C Bus is always initialized and configured in SDR Mode, never in any of
395 the HDR Modes. (The procedure for entering an HDR Mode from SDR Mode is detailed in *Section 5.2*.)

396 As a result, most of the essential basic I3C protocol specifications are found in *Section 5.1*, including:

397

Subject	Section
Bus Configuration	5.1.1
Bus Communication	5.1.2
Bus Free Condition	5.1.3.2
Bus Idle Condition	5.1.3.4
Bus Initialization and Dynamic Address Assignment Mode	5.1.4
Hot-Join Mechanism	5.1.5
In-Band Interrupt	5.1.6
Secondary Master Functions	5.1.7
Timing Control	5.1.8
Common Command Codes (CCC)	5.1.9

5.1 Single Data Rate (SDR) Mode

398 This Section specifies the communication protocols for Single Data Rate (SDR) Mode.

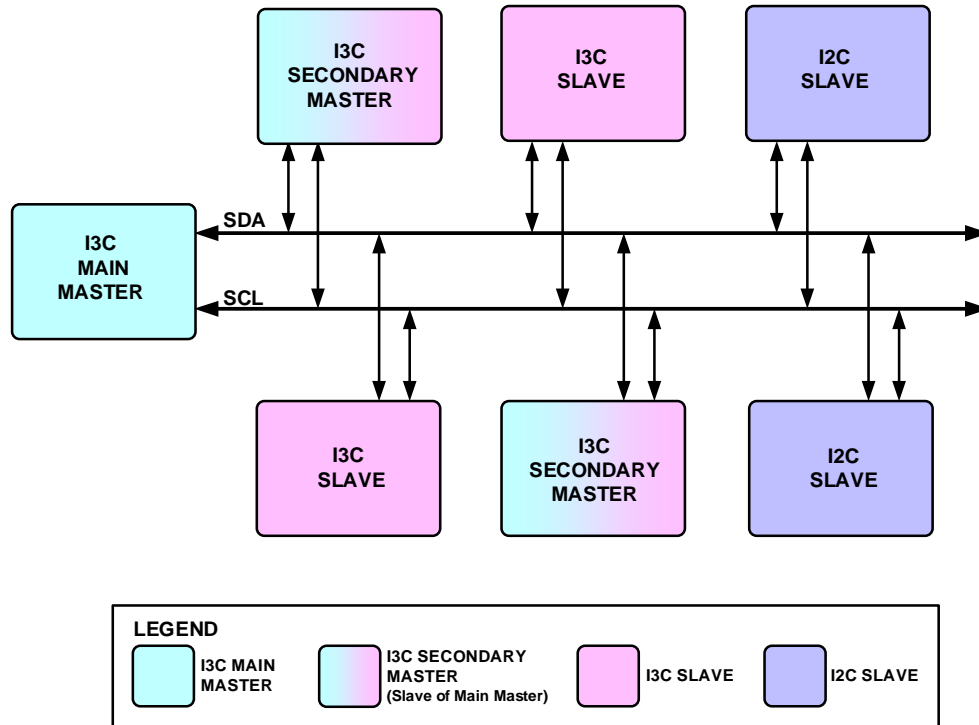
399 SDR Mode is the default Mode of the I3C Bus, and is primarily used for private messaging from the Current
400 Master Device to Slave Devices. SDR Mode is also used to enter other Modes, sub-Modes, and states (as
401 described in *Section 5.1* and *Section 5.2*); and for built-in features such as Common Commands (CCCs), In-
402 Band Interrupts, and transition from I²C to I3C by assignment of a Dynamic Address.

403 I3C SDR Mode is significantly similar to the I²C protocol [*NXP01*] in terms of procedures and conditions,
404 and as a result I3C Devices and many Legacy I²C Slave Devices (but not I²C Master Devices) can coexist on
405 the same I3C Bus. However SDR Mode also includes numerous new features not present in I²C. For the
406 procedures and conditions that I3C shares with I²C, SDR Mode closely follows the definitions in the I²C
407 Specification. I²C traffic from an I3C Master to an I²C Slave will be properly ignored by all I3C Slaves,
408 because the I3C protocol is designed to allow I²C traffic. I3C traffic from an I3C Master to an I3C Slave will
409 not be seen by most Legacy I²C Slave Devices, because the I²C Spike Filter is opaque to I3C's higher clock
410 speed.

5.1.1 Bus Configuration

411 The I3C Bus can be configured as the link among several clients, in a flexible and efficient manner. At the
412 system architecture level, eight roles are defined for I3C compatible Devices (see *Table 2*).

413 An example block diagram of I3C interconnections is shown in *Figure 8*. In this diagram the color blue
414 indicates Devices with a Master role, the color pink indicates Devices with an I3C Slave role, and the color
415 purple indicates Devices with an I²C Slave role. Note that I3C Secondary Master Devices are shaded from
416 blue to pink, illustrating their ability to function in both Master and Slave roles (at different times).



417
418

Figure 8 I3C Bus with I²C Devices and I3C Devices

419 I3C compatible Devices may have diverse features, as appropriate for their function within the I3C Bus.
420 Depending on the I3C Bus' system design, it may not be necessary for all features of a given Device to be
421 enabled for any particular Bus instantiation. However, the enabled features of every I3C compatible Device
422 shall be described in Characteristics Registers associated with the Device, as described in *Section 5.1.1.2*.
423 The I3C Main Master shall obtain the Characteristics of any Legacy I²C Devices on the I3C Bus before power
424 up (e.g. the fixed Address of each Legacy I²C Device present on the Bus).

425 At every start-up from a powered-down state, the Main Master shall assign a unique Dynamic Address to
426 every Device on the Bus, including itself. The Dynamic Address assignment procedure is described in
427 *Section 5.1.4*. Dynamic Addresses create a priority ranking of the Devices' In-Band Interrupts. Any
428 Secondary Masters present on the I3C Bus shall be made aware of the Dynamic Address assignments and
429 Characteristic Registers associated with each I3C compatible Device on the Bus via Common Command
430 Codes as described in *Section 5.1.9*.

5.1.1.1 I3C Device Characteristics

431 The configuration of an I3C Bus will depend upon the Characteristics of the I3C Devices intended to be
432 active on that I3C Bus. Therefore, an active I3C Device playing a given Role in a given I3C Bus instantiation
433 shall fulfill all responsibilities for that Role, as detailed in *Table 3*.

434

Table 3 I3C Devices Roles vs Responsibilities

Responsibilities / Features	Comments	Roles					
		Main Master	Secondary Master	SDR Only Main Master	SDR Only Secondary Master	Slave	SDR Only Slave
Manages SDA Arbitration	For Address Arbitration, In-Band Interrupt, Hot-Join, Dynamic Address, as appropriate	Y	Y	Y	Y	N	N
Dynamic Address Assignment	Master assigns Dynamic Address	Y	N	Y	N	N	N
Hot-Join Dynamic Address Assignment	Master capable of assignment Dynamic Address after Hot-join	Y	Optional	Y	Optional	N	N
Self Dynamic Address Assignment	Only Main Master can self-assign a Dynamic Address	Y	N	Y	N	N	N
Static I2C Address¹	–	N/A	Optional	N/A	Optional	Optional	Optional
Memory for Slaves' Addresses and Characteristics	Retaining registers	Y	Y	Y	Y	N	N
HDR Slave capable	Supports being accessed in at least one HDR Mode	Y	Y	N	N	Y	N
HDR Master capable	Supports Mastering in at least one HDR Mode	Y	Y	N	N	N/A	N/A
HDR Exit Pattern Generation capable²	Able to generate the HDR Exit Pattern on the Bus for error recovery	Y	Y	Y	Y	N	N
HDR Tolerant	Recognizes HDR Exit Pattern	Y	Y	Y	Y	Y	Y
Note:							
1) A Static Address may be used to more quickly assign a Dynamic Address. See Section 5.1.4 .							
2) All Slaves require an HDR Exit Pattern Detector, even Slaves that are not HDR capable							

435

436 The I3C protocol supports a subset of I²C Slave features. For example, an I3C Slave can have a Static Address
437 but also support Dynamic Addressing. A Device shall not have a 50 ns filter enabled when used in an I3C
438 Bus operating at full clock speed. These differences are summarized in **Table 4**. When used in an I3C system,
439 I3C Slaves shall enable or disable appropriate I²C features as shown in **Table 4**.

440

Table 4 I²C Features Allowed in I3C Slaves

I ² C Feature When Used on an I3C Bus	Required on I3C	Desirable on I3C	Not Used on I3C	Not Allowed on I3C	Note
Fm Speed	–	–	X	–	3
Fm+ Speed	X	–	–	–	–
HS Speed	–	–	X	–	3
UFm Speed	–	–	X	–	3
Static I ² C Address	–	X	–	–	–
50ns Spike Filter	–	–	X	X (Shall disable)	3
Clock Stretch	–	–	–	X	–
20mA Open Drain Driver	–	–	X	–	1, 3
Matches I ² C AC Timing	–	–	X	–	2, 3
I ² C Extended Address (10 bit)	–	–	X	–	3
I3C Reserved Addresses	–	–	–	X	–
Note:					
1) See Table 71 and Table 73					
2) I3C drive and timing requirements are different from I ² C					
3) If an I3C Slave has I ² C features intended for use on an I ² C Bus, then they will not be used on an I3C Bus. As stated in Section 5.1.1.1 , once the Slave sees a 7'h7E, it will disable I ² C features that are not used by I3C.					

441 Performance of an I3C Bus is heavily dependent upon any I²C-only Devices that may be connected to that
442 Bus. Consequently, all I²C-only Devices permitted on any instantiation of an I3C Bus must be compliant with
443 one of the categories detailed in **Table 5**. Furthermore (and as referenced in **Table 72**), no I²C or I3C Device
444 present on an I3C Bus shall have a fixed I²C Address that matches any of the Addresses associated with Error
445 Type S0 (see **Table 59**).

446

Table 5 Legacy I²C-Only Slave Categories and Characteristics

Index Specific	I ² C-Only Devices Index 0	I ² C-Only Devices Index 1	I ² C-Only Devices Index 2
50ns IO Spike Filter ¹	Y	N	N
Max SCL clock frequency (f _{SCL}) tolerant ²	N/A	Y	N
Note:			
1) Allows tolerance of HDR Modes and SDR at SCL High periods of t _{DIG_H_MIXED} or less			
2) Allows compliance up to maximum SDR SCL clock frequency (f _{SCL})			

5.1.1.2 I3C Characteristics Registers

447 I3C Characteristics Registers describe and define an I3C compatible Device's capabilities and functions on
448 the I3C Bus, as the Device services a given system. Devices without I3C Characteristics Registers shall not
449 be connected to a common I3C Bus.

450 There are three Characteristics Register types:

- 451 · **Bus Characteristics Register** (BCR, see *Section 5.1.1.2.1*)
- 452 · **Device Characteristics Register** (DCR, see *Section 5.1.1.2.2*)
- 453 · **Legacy Virtual Register** (LVR, see *Section 5.1.1.2.3*)

454 Every I3C compatible Device shall have associated Characteristics Registers, depending on the Device type
455 as described below:

- 456 · Every I3C compliant Device (as defined in *Table 3*) shall have one Bus Characteristics Register,
457 and one Device Characteristics Register.
- 458 · Every Legacy I²C Device to be connected to an I3C Bus shall have one associated Legacy Virtual
459 Register. Since these are Legacy Devices, it is understood that this register will exist virtually, for
460 example as part of the Device's driver.

5.1.1.2.1 Bus Characteristics Register (BCR)

461 Each I3C Device that is connected to the I3C Bus shall have an associated read-only Bus Characteristics
462 Register (BCR). This read-only register describes the I3C compliant Device's role and capabilities for use in
463 Dynamic Address assignment and Common Command Codes. The bits within the BCR shall conform to the
464 Descriptions presented in in **Table 6**.

465

Table 6 Bus Characteristics Register (BCR)

BIT	Name	Description
BCR [7]	Device Role [1]	2'b00 – I3C Slave 2'b01 – I3C Master ¹
BCR [6]	Device Role [0]	2'b10 – Reserved for future definition by MIPI Sensor WG 2'b11 – Reserved for future definition by MIPI Sensor WG
BCR [5]	SDR Only / SDR and HDR Capable	0 – SDR only 1 – HDR Capable
BCR [4]	Bridge Identifier ²	0 – Not a Bridge Device 1 – Is a Bridge Device
BCR [3]	Offline Capable ³	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, as described in Section 5.1.2.3.3
BCR [1]	IBI Request Capable	0 – Not Capable 1 – Capable
BCR [0]	Max Data Speed Limitation ⁴	0 – No Limitation 1 – Limitation
<p>Note:</p> <p>1) For an I3C Device acting as I3C Main Master, the BCR Device Role bits will contain the value 2'b01.</p> <p>2) Bridge Devices are required to comply with this MIPI Specification for I3C.</p> <p>3) Offline Capable Devices retain the Dynamic Address, and are specified in Section 2.2.</p> <p>4) Master shall use the GETMXDS CCC to interrogate the Slave for specific limitation.</p>		

5.1.1.2.2 Device Characteristics Register (DCR)

466 Each I3C Device that is connected to the I3C Bus shall have an associated read-only Device Characteristics
 467 Register (DCR). This read-only register describes the I3C compliant Device type (e.g. accelerometer,
 468 gyroscope, etc.) for use in Dynamic Address assignment and Common Command Codes. The bits within the
 469 DCR shall conform to the Descriptions presented in *Table 7*.

470 **Table 7 I3C Device Characteristics Register (DCR)**

Bit	Name	Description
DCR [7]	Device ID [7]	255 available codes for describing the type of sensor, or Device. Examples: Accelerometer, gyroscope, composite Devices. Default value is 8'b0: Generic Device
DCR [6]	Device ID [6]	
DCR [5]	Device ID [5]	
DCR [4]	Device ID [4]	
DCR [3]	Device ID [3]	
DCR [2]	Device ID [2]	
DCR [1]	Device ID [1]	
DCR [0]	Device ID [0]	

5.1.1.2.3 Legacy Virtual Register (LVR)

471 Each Legacy I²C Device that can be connected to the I3C Bus shall have an associated read-only Legacy
 472 Virtual Register (LVR) describing the Device's significant features. Since these are Legacy I²C Devices, it is
 473 understood that this register will exist virtually, for example as part of the Device's driver. When Legacy I²C
 474 Devices are present on an I3C Bus, LVR data determines allowed Modes and maximum SCL clock frequency.
 475 The bits within the LVR shall conform to the descriptions in *Table 8*.

476 All LVRs shall be established by the higher-level entity controlling the I3C Bus, and transferred to the I3C
 477 Bus Main Master prior to Bus configuration. The LVR content for all I²C Devices is always known by the
 478 Main Master. The LVR information can be transferred to Secondary Masters by using the DEFSLVS CCC
 479 (see *Section 5.1.9.3.7*).

480 **Table 8 Legacy I²C Virtual Register (LVR)**

Bit	Name	Description
LVR [7]	Legacy I ² C only [2] (Indexed in <i>Table 5</i>)	3'b000 – Index 0 3'b001 – Index 1 3'b010 – Index 2
LVR [6]	Legacy I ² C only [1] (Indexed in <i>Table 5</i>)	3'b011 – Index 3 (Reserved) 3'b100 – Index 4 (Reserved) 3'b101 – Index 5 (Reserved)
LVR [5]	Legacy I ² C only [0] (Indexed in <i>Table 5</i>)	3'b110 – Index 6 (Reserved) 3'b111 – Index 7 (Reserved)
LVR [4]	I ² C Mode Indicator	0 – I2C Fm+ 1 – I2C Fm
LVR [3]	MIPI Sensor WG Reserved	15 available codes for describing the Device capabilities and function on the sensors' system.
LVR [2]	MIPI Sensor WG Reserved	
LVR [1]	MIPI Sensor WG Reserved	
LVR [0]	MIPI Sensor WG Reserved	

5.1.2 Bus Communication

481 The primary protocol and Mode of I3C is SDR (Single Data Rate) Mode. The SDR protocol is based on the
482 I²C standard protocol [NXP01], with a few notable variations:

- 483 • The I3C START and STOP (as shown in *Figure 66* and *Figure 69*, respectively) are identical to the I²C
484 START and STOP in their signaling, but they may vary from I²C in their timing. Compare *Table 74* vs.
485 *Table 73*.
- 486 • The I3C Address Header is identical to the I²C Address Header in bit form and in signaling, but it may
487 vary from I²C in its timing. See *Section 5.1.2.2*, *Table 74*, and *Table 75*.
- 488 • The Data 9-bit Words use the same bit count as I²C, but differ in the ninth bit, as explained in
489 *Section 5.1.2.3*.
- 490 • In I3C, the SCL line is only driven by the Master. Normally this drive is Push-Pull, but it can also be
491 Open Drain.

492 Because the bit count is the same for Address Header and Data Word, the I3C Slave only needs to know
493 whether a Message is an I3C Messages (vs. is an I²C Message) if the Message is addressed to that Slave
494 (either directly or by Broadcast).

495 An I3C Message is defined as everything from the initial START (or Repeated START) to the next Repeated
496 START or STOP.

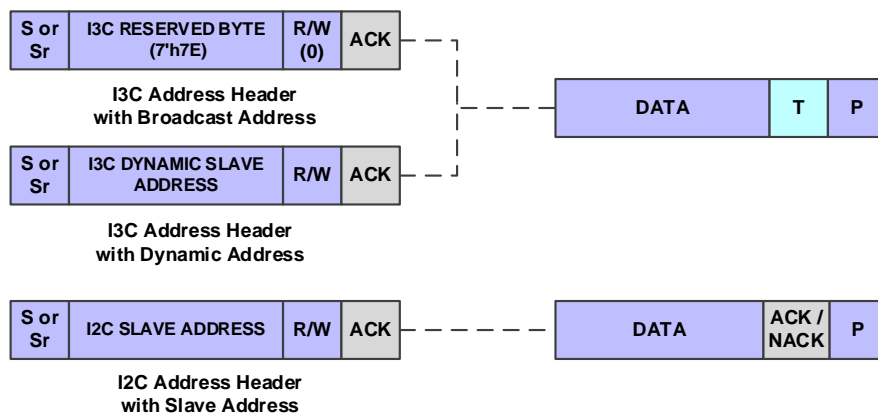
497 An I3C Message is an SDR Message if:

- 498 • The Address in the Address Header is 7'h7E (the I3C Broadcast Address). All I3C Slaves shall match
499 Address value 7'h7E. No I²C Slave will match Address 7'h7E, because that value is reserved and unused
500 in I²C.
- 501 • The Address in the Address Header matches the Slave's Dynamic Address (as assigned by the I3C
502 Master per *Section 5.1.4*). All I3C Slaves shall match their own Dynamic Address. (It is permitted to
503 then NACK the Header if needed.)

504 All I3C Slaves shall ignore all Messages with Addresses other than 7'h7E or the I3C Master Assigned
505 Address, and await either the Repeated START or the STOP. I3C Slaves shall not transmit on the Bus in
506 response to a non-matching Address.

507 **Note:**

508 *Legacy I²C Slaves will ignore any Message not addressed to them, and will await the next START or*
509 *STOP. Per **Section 5.1.2.4**, Legacy I²C Slaves may also not see some or all of I3C Messages and*
510 *Modes due to the speed of SCL signaling.*



511
512

Figure 9 Address Header Comparison

5.1.2.1 Role of I3C Slave

513 The I3C Slave does not have to know whether it is on a Legacy I²C Bus or an I3C Bus. If it has a Legacy I²C
514 Static Address, then it may participate using that Address up until (and if) it is assigned a Dynamic Address.
515 Once assigned a Dynamic Address, unless asked to Reset, it shall only operate as an I3C Slave.

516 Before being assigned a Dynamic Address the I3C Slave shall operate as an I²C Device, with the following
517 added features:

- 518 · The I3C Slave shall ACK the I3C Broadcast Address (7'h7E), if assessed only immediately after
519 START condition.
- 520 · The I3C Slave shall process the CCCs ENTDA (see *Section 5.1.9.3.4*) and SETDASA (see
521 *Section 5.1.9.3.10*), as appropriate.
- 522 · The I3C Slave shall ignore all S0 type errors (see *Table 59*).

523 The role of an I3C Slave shall be as follows:

- 524 1. Following a START or a Repeated START, at any speed conforming to *Section 6* of this
525 Specification, the I3C Slave shall attempt to match the Address to the I3C Broadcast Address
526 (7'h7E) or to its own Dynamic Address, once assigned. If a match is found, then the I3C Slave
527 shall treat that Message as I3C SDR.
- 528 2. If the Message is addressed to the Slave's Dynamic Address, then the Slave may ACK or passively
529 NACK the Address Header:
 - 530 a. If the Slave ACKs the Address Header, then the Slave shall process the Message as I3C SDR,
531 following all rules as outlined in this Section.
 - 532 b. If the Slave NACKs the Address Header (does not drive the ACK bit Low), then the Slave
533 may disregard any bits that follow, up until the next Repeated START or STOP.
- 534 3. If the Message is addressed to the I3C Broadcast Address (7'h7E), with a Write (RnW bit is 0),
535 then the Slave shall process that Message at least through the first byte of data (if any data is
536 present in the Message):
 - 537 a. If there is a byte of data in a 7'h7E Broadcast Message, then the Message is a CCC (Common
538 Command Code) Command, per *Section 5.1.9*.
 - 539 b. The I3C Slave shall process all applicable Required CCC commands, per *Section 5.1.9.3*. A
540 command may be either Always Required, or only Contextually Required, per *Table 9*.
 - 541 c. If the CCC command changes the Mode of the I3C Bus, then the I3C Slave shall handle the
542 new Mode in one of two ways.

543 **Either:**

- 544 i. If the new Mode is Dynamic Address Assignment Mode (see *Section 5.1.4*), and required
545 for all I3C Slaves, then the Slave shall participate if it does not have a current Dynamic
546 Address; otherwise, the Slave shall await the STOP that indicates exit from Dynamic
547 Address Assignment Mode.

548 **Or:**

- 549 ii. If the new Mode is HDR (High Data Rate) Mode, then the Slave may either enter into
550 HDR Mode if it supports that specific HDR (per *Section 5.2.2* and *Section 5.2.3*), or else
551 enable its HDR Exit Pattern detector (per *Section 5.2.1* and *Section 5.2.1.3*) to await the
552 exit from HDR Mode.
- 553 4. If the Message is not addressed to the I3C Broadcast Address (7'h7E) or to the Slave's Dynamic
554 Address, then the I3C Slave shall await either a Repeated START or a STOP. The Slave may
555 record/monitor the bits as they pass (if desired), but the only obligation is to wait for either a
556 Repeated START or a STOP:
 - 557 · A Repeated START is defined by the SDA line changing from High to Low while the SCL line
558 is High, per *Section 6*.

- 559 · A STOP is indicated by the SDA line changing from Low to High while the SCL line is High,
560 per *Section 6*.
561 In both cases, I3C timing may or may not be the same as with I²C.

5.1.2.2 I3C Address Header

562 The I3C Address Header follows either a START, or a Repeated START. The format is the same as I²C: 7
563 bits of Address, 1 bit of RnW, and 1 bit of ACK/NACK.

564 The Address Header following a START is an Arbitrable Address Header, as explained in *Section 5.1.2.2.1*.
565 This means the START and at least the first Address bit and ACK/NACK are issued on SDA using Open
566 Drain Bus drive, similar to I²C. However, some of the Arbitrable Address Header may be driven on SDA
567 using Push-Pull and higher speed (see *Section 5.1.2.2.2*).

568 The Address Header following a Repeated START is always driven on SDA using Push-Pull, with the
569 exception of the ACK/NACK (see *Section 5.1.2.2.4*).

570 Using the I3C Arbitrable Address Header, I3C Slaves may transmit any of three requests to the I3C Master:

- 571 1. An In-Band Interrupt, per *Section 5.1.6*. This is equivalent to toggling a wire to get the Master's
572 attention. The In-Band Interrupt request shall be made using the Slave's Dynamic Address with a
573 RnW bit of 1.
- 574 2. A Secondary Master request, per *Section 5.1.7*. An I3C Slave shall not make such a request unless
575 it is marked as a Secondary Master in its BCR register, per *Section 5.1.1.2.1*. The Secondary
576 Master request shall be made using the Slave's Dynamic Address with a RnW bit of 0.
- 577 3. A Hot-Join request, per *Section 5.1.5*. An I3C Slave shall only make such a request when
578 becoming available after the I3C Bus is operational. The Hot-Join request shall be made using the
579 special Hot-Join Address of 7'h02.

580 The I3C Slaves shall make these requests to the I3C Master in only two Bus conditions:

- 581 1. A START (but not a Repeated START) is issued on the Bus following a Bus Available Condition,
582 per *Section 5.1.3*. The Slave may transmit its Dynamic Address or the Hot-Join Address (7'h02)
583 following the START, by adhering to the I3C Address Arbitration rules per *Section 5.1.2.2.1*.
- 584 2. The Bus is in a Bus Available Condition, per *Section 5.1.3*, so the Slave may issue a START by
585 pulling the SDA Low.
 - 586 a. If the Slave pulls the SDA Low, then the Master shall pull SCL Low within a best-efforts
587 period of time, where that time is not explicitly defined.
 - 588 b. The Master shall also pull SDA Low (overlapping the Slave pulling it Low).
 - 589 c. Once the Master has pulled the SCL Low, the Slave shall control the SDA line in Open Drain
590 mode (i.e., either pull Low, or release High).
 - 591 d. The Slave may then issue its Address in the normal way (condition 1 above).

5.1.2.2.1 I3C Address Arbitration

592 An Address Header following a START (but not a Repeated START) is subject to Arbitration, meaning both
593 the Master and one or more Slaves may attempt to drive an Address onto the Bus, using SDA. Such Address
594 Headers are defined as Arbitrable Address Headers.

595 The Arbitration model follows the common Open Drain approach. All Devices (whether Master or Slave)
596 that are transmitting an Address shall then follow the same rule:

- 597 1. If the current bit to transmit is a 0, then the Device shall drive SDA Low after the falling edge of
598 SCL and hold Low until the next falling edge of SCL.

599 **Note:**

600 *Other Devices may also be driving SDA Low, but that is acceptable.*

- 601 2. If the current bit to transmit is a 1, then the Device shall not drive SDA, but rather shall High-Z
602 SDA on the falling edge of SCL.
 - 603 a. Additionally, the Device shall monitor the SDA on the rising edge of SCL to determine
604 whether another Device has driven SDA Low.
 - 605 b. If another Device has driven the SDA Low, then the Device has “lost” the Arbitration and
606 shall not further participate in this Address Header. That is, the Device shall not transmit any
607 more bits, but may wait for a future START Condition (but not a Repeated START
608 Condition).

5.1.2.2.2 I3C Address Arbitration Optimization

609 I3C Address Arbitration may optionally be optimized, as detailed in this Section.

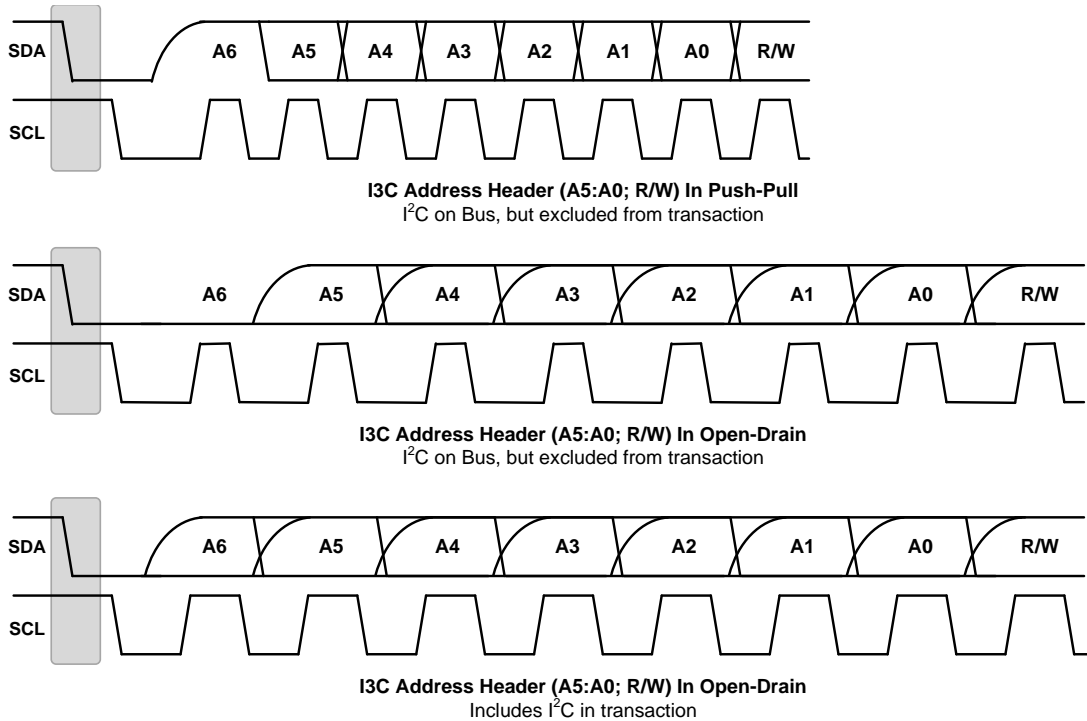
610 As previously described in *Section 5.1.2.2*, an I3C Master Device assigns 7-bit Dynamic Addresses with
611 values in the range 7'h03 to 7'h7B. However because the I3C Master treats the entire 9-bit Arbitrable Address
612 Header as Open Drain, it has no way to detect whether a Slave Device might be transmitting its own Address
613 during some (or all) of the Address Header.

614 Note that for I3C Secondary Masters and I3C In-Band Interrupt Slaves, the I3C Master is free to restrict
615 assigned Dynamic Addresses to the lower half of the available range (7'h03 to 7'h3F), thus leaving the value
616 of Address bit A6 (the first Address bit after the START) as value 0 in all assigned Dynamic Addresses.

617 Having restricted Dynamic Addresses in this manner, the I3C Master may then optionally optimize the
618 Arbitrable Address Header as follows:

- 619 • If the I3C Master is transmitting a 1 value (i.e. High-Z on SDA), as it would when transmitting
620 7'h7E, then it can monitor SDA on the rising edge of SCL. If SDA has the value 1 (i.e. if SDA is
621 not being driven by any Slave), then the I3C Master may optionally transmit the remainder of the
622 Address Header (up until the ACK/NACK) in Push-Pull mode. See *Figure 10*, upper waveform.
- 623 • If the I3C Master is transmitting a 0 value (i.e. is driving SDA Low), or if SDA was driven Low by
624 a Slave, then the I3C Master shall transmit the remainder of the Address Header using Open Drain.
- 625 • If the I3C Master intends to transmit solely to other I3C Devices (i.e. not to any I²C Devices), then
626 it may optionally maintain the SCL pulse width below 50ns, with the result that any I²C Devices
627 present on the Bus will see only a 0 value. This shorter pulse width produces a higher data rate,
628 because for Open Drain Arbitration only the Low time is extended. See *Figure 10*, middle
629 waveform.
- 630 • If the I3C Master intends to transmit to any I²C Devices, then it must use the slower I²C timing.
631 See *Figure 10*, lower waveform.

632 The upper waveform of *Figure 10* illustrates this optimization. When the I3C Master sees a value of 1 for
633 Address bit A6, but previously made sure that all Dynamic Addresses assigned to I3C In-Band Interrupt
634 Slaves have a 0 in bit A6, then the I3C Master knows that the line was not being driven by any such Slave.



635
636

Figure 10 Address Arbitration During Header

5.1.2.2.3 Consequence of Master Starting a Frame with an I3C Slave Address

637 The I3C Master normally should start a Frame with 7'h7E (for all I3C Messages) or an I²C Static Address
638 (when sending only to a Legacy I²C Slave).

639 In both cases the Address may be arbitrated, and so the Master shall monitor to see whether an IBI, Mastership
640 request (Slave requests to become the Master), or Hot-Join request has been made.

- 641 · If not, then the Master may proceed normally.
- 642 · If so, then the Master may ACK or NACK that request and then proceed accordingly.

643 If the Master chooses to start an I3C Message with an I3C Dynamic Address, then special provisions shall
644 be made because that same I3C Slave may be initiating an IBI or Mastership request. So, one of three things
645 may happen:

- 646 1. The Addresses match, but the difference is caught on the RnW bit, and the Master was Writing
647 (RnW=0); so the Master wins (IBI with RnW=1 loses), and proceeds normally.
- 648 2. The Addresses match, but the difference is caught on the RnW bit, and the Master was Reading
649 (RnW=1); so the Master loses, and must ACK or NACK the Mastership request (RnW=0).
- 650 3. The Addresses match as do the RnW and so neither Master nor Slave shall ACK since both are
651 expecting the other side to.
 - 652 a. This is a problem since the Master cannot tell whether the NACK was due to this condition,
653 or simply because the Slave may have chosen to NACK it.
 - 654 b. The Master shall transmit the Slave Address again after a Repeated START (the next one or
655 any one prior to a STOP in the Frame). This allows it to determine which condition occurred
656 and to avoid a deadlock.

5.1.2.2.4 Address Header Following a Repeated START is Push-Pull

657 The Address transmitted by the I3C Master following a Repeated START shall not be arbitrated. That is, no
658 I3C Slave shall attempt to transmit its own Dynamic Address nor the Hot-Join Address following a Repeated
659 START.

660 As a result, the Address Header (i.e., the 7 bits of Address plus the RnW bit) shall be transmitted on SDA
661 using Push-Pull mode when the Message is not to a Legacy I²C Slave. The ACK/NACK bit that follows the
662 RnW bit is always Open Drain to allow the Slave to ACK or passively NACK its Address.

5.1.2.2.5 I3C Slave Address Restrictions

663 The I3C Slave Address space is dependent on decisions by the Master. That is, the Master may choose the
664 Dynamic Addresses from a set of values, observing optional and non-optional restrictions as follows. These
665 restrictions are also illustrated in *Table 9*.

- 666 · The I3C Master shall not use any of 7'h00, 7'h01, 7'h02, 7'h7E, 7'h7F. All are reserved for I3C.
- 667 · The I3C Master shall not use any of 7'h3E, 7'h5E, 7'h6E, 7'h76, 7'h7A, 7'h7C, 7'h7F. All are
668 prohibited for detecting an error in the Broadcast Address (7'h7E).
- 669 · The I3C Master may choose to not use 7'h03, marked in I²C as reserved.
- 670 · The I3C Master shall not use 7'h04, 7'h05, 7'h06, 7'h07 if any Legacy I²C Devices are present on
671 the Bus that support I²C “High-Speed Mode”.
- 672 · The I3C Master shall not use 7'h7C or 7'h7D if any Legacy I²C Devices are present on the Bus
673 that support I²C Device ID Mode.
- 674 · The I3C Master shall not use 7'h78, 7'h79, 7'h7A, 7'h7B if any Legacy I²C Devices are present on
675 the Bus that support I²C Extended Address Mode and have an Extended Address or would be
676 impacted by the Extended Address mechanism.

677

Table 9 I3C Slave Address Restrictions

Slave Dynamic Address		Restriction	Description
Binary	Hex		
000 0000	7'h00	Shall not use	I3C Reserved
000 0001	7'h01	Shall not use	I3C Reserved: For use with SETDASA CCC in special Point-to-Point Communication. See Section 5.1.9.3.10 .
000 0010	7'h02	Shall not use	I3C Reserved: Hot-Join Address
000 0011	7'h03	Optional	Marked 'Reserved' by I ² C
000 0100	7'h04	Conditional	Available for use only if no Legacy I ² C Devices supporting I ² C "High-Speed Mode" are present on the Bus
000 0101	7'h05		
000 0011	7'h06		
000 0011	7'h07		
000 1000 011 1101	7'h08 – 7'h3D	Available for use	54 Addresses
011 1110	7'h3E	Shall not use	I3C Reserved: Broadcast Address single bit error detect
011 1111 101 1101	7'h3F – 7'h5D	Available for use	31 Addresses
101 1110	7'h5E	Shall not use	I3C Reserved: Broadcast Address single bit error detect
101 1111 110 1101	7'h5F – 7'h6D	Available for use	15 Addresses
110 1110	7'h6E	Shall not use	I3C Reserved: Broadcast Address single bit error detect
110 1111 111 0101	7'h6F – 7'h75	Available for use	7 Addresses
111 0110	7'h76	Shall not use	I3C Reserved: Broadcast Address single bit error detect
111 0111	7'h77	Available for use	1 Address
111 1000	7'h78	Conditional	Available for use only if no Legacy I ² C Devices are present on the Bus that both a) support I ² C "Extended Address Mode", and b) either have an Extended Address, or would be impacted by the Extended Address mechanism
111 1001	7'h79		
111 1010	7'h7A	Shall not use	I3C Reserved: Broadcast Address single bit error detect
111 1011	7'h7B	Conditional	Available for use only if no Legacy I ² C Devices are present on the Bus that both a) support I ² C "Extended Address Mode", and b) either have an Extended Address, or would be impacted by the Extended Address mechanism
111 1100	7'h7C	Shall not use	I3C Reserved: Broadcast Address single bit error detect (Also not available for use if any Legacy I ² C Devices supporting I ² C "Device ID Mode" are present on the Bus.)
111 1101	7'h7D	Conditional	Available for use only if no Legacy I ² C Devices supporting I ² C "Device ID Mode" are on the Bus
111 1110	7'h7E	Shall not use	I3C Reserved: Broadcast Address
111 1111	7'h7F	Shall not use	I3C Reserved: Broadcast Address single bit error detect

5.1.2.3 I3C SDR Data Words

678 In I3C SDR, the Data Words match I²C only in the sense that they are both 9 bits long. I3C SDR Data Words
679 differ from I²C in three ways, as detailed in *Sub-Sections 5.1.2.3.1, 5.1.2.3.2, and 5.1.2.3.3.*

680 In summary:

- 681 1. **Handoff from Address ACK to SDR Master Write Data:** When performing an SDR Write, the
682 handoff from the Slave's Address Header ACK to the Master's first Data bit is different in I3C. I²C
683 is Open Drain, so overlap from the ACK Low into the first bit is harmless. By contrast, I3C is
684 Push-Pull and so this handoff is specified (see *Section 5.1.2.3.1*).
- 685 2. **Ninth Bit of SDR Master Written Data as Parity:** In I²C, the ninth Data bit written by the
686 Master is an ACK by the Slave. By contrast, in I3C the ninth Data bit written by the Master is the
687 Parity of the preceding eight Data bits. Therefore, in I3C the Slave shall not drive the SDA line for
688 Data written by the Master in SDR. In SDR terms, the ninth bit of Write data is referred to as the
689 T-Bit (for 'Transition') (see *Section 5.1.2.3.2*).
- 690 3. **Ninth Bit of SDR Slave Returned (Read) Data as End-of-Data:** In I²C, the ninth Data bit from
691 Slave to Master is an ACK by the Master. By contrast, in I3C this bit allows the Slave to end a
692 Read, and allows the Master to Abort a Read. In SDR terms, the ninth bit of Read data is referred
693 to as the T-Bit (for 'Transition') (see *Section 5.1.2.3.3*).

5.1.2.3.1 Transition from Address ACK to SDR Master Write Data

694 The end of any Address Header (whether Arbitrated or not) is an ACK or NACK by the one or more addressed
695 Slaves, using Open Drain on SDA:

- 696 · If 7'h7E, then it is the ACK of all I3C Slaves on the Bus.
- 697 · If a single Slave Address, then it is the ACK (or NACK) of the addressed Slave, or a NACK if no
698 such Slave is on the Bus.

699 When the Address Header results in an ACK, and the Message is SDR Write from Master, the SDA line has
700 to be turned from Open Drain to Push-Pull for the first data bit. To do that safely, I3C SDR specifies how the
701 handoff is to occur. This is summarized below and shown in *Figure 61*.

- 702 1. The I3C Slave shall hold the SDA line Low during the ACK (while SCL is Low).
703 · This is an Open Drain SCL Low period.
- 704 2. After the I3C Slave sees the rising edge of SCL, it releases the SDA line to High-Z.
705 · The I3C Slave shall release the SDA line using normal (Push-Pull) timing (release the SDA line
706 as soon as it sees SCL rising).
- 707 3. After the rising edge of SCL, the I3C Master shall drive the SDA line Low.
708 · As a result, both Master and Slave will be driving the SDA line Low for a short overlap (which
709 is safe).
710 · The SCL High period may be as short as the minimal t_{DIG_H} , per *Section 6.2*.
- 711 4. On the falling edge of SCL the I3C Master shall begin driving data on the SDA line, using Push-
712 Pull drive as shown in *Figure 61*.

713 When the Address Header results in a NACK, the Master may choose to either:

- 714 1. Continue the transaction, by generating a Repeated START
- 715 **or**
- 716 2. Relinquish the Bus, by generating a STOP as shown in *Figure 62*.

5.1.2.3.2 Ninth Bit of SDR Master Written Data as Parity

717 The ninth data bit of each SDR Data Word written by the I3C Master (also referred to as the T-Bit) is a Parity
718 bit, calculated using odd parity. Parity can help in detecting noise-caused errors on the line. The value of this
719 Parity bit shall be the XOR of the 8 Data bits with 1, i.e.: XOR(Data [7:0], 1).

720 Examples:

- 721 · If all eight data bits are 1's (0xFF), then the Parity bit value will be 1.
- 722 · If all eight data bits are 0's (0x00), then the Parity bit value will be 1.
- 723 · If an odd number of bits in the Data Word are 1's (e.g. 0xFE or 0x01), then the Parity bit value will
724 be 0.

725 T (Parity) bit writes shall always be kept valid through the SCL High period. In the case of a T-Bit
726 representing the last data byte, the write is therefore kept valid through the SCL High period, and the next
727 SCL Low can then be used to either change the SDA, or not change the SDA, in preparation for the Repeated
728 START or STOP that follows.

5.1.2.3.3 Ninth Bit of SDR Slave Returned (Read) Data as End-of-Data

729 In I²C, Read from Slave has the issue that only the Master ends the Read, so the Slave has no ability to control
730 the amount of data it returns. In I3C SDR, by contrast, the Slave controls the number of data Words it returns;
731 but it also allows the I3C Master to abort the Read prematurely when necessary.

732 This mechanism is controlled purely by the ninth (T) Data bit of each SDR Data Word returned by the I3C
733 Slave. The ninth bit is returned by the Slave in one of three ways, as explained below.

- 734 · The I3C Slave returns the ninth bit as 0 (SDA Low) to end the Message:
 - 735 · The Slave shall set SDA Low on the falling edge of SCL.
 - 736 · On the following rising edge of SCL, the Slave shall set SDA to High-Z.
 - 737 · The I3C Master shall drive SDA Low on the rising edge of SCL, thereby overlapping with the
738 Slave.
 - 739 · The I3C Master then shall issue either a STOP as shown in *Figure 69*, or a Repeated START as
740 shown in *Figure 70* (on the next clock, or one after, per the normal I²C procedure for setting
741 up SDA to issue a Repeated START).
- 742 · The I3C Slave returns the ninth bit as 1 (SDA High) to continue the Message (and permit the
743 Master to abort the Message):
 - 744 · The Slave shall set SDA High on the falling edge of SCL.
 - 745 · On the following rising edge of SCL, the Slave shall set SDA to High-Z, thereby Parking the
746 Bus for the SCL High period:
 - 747 · If the I3C Master is able to continue the reply from the Slave, then it shall do nothing. The
748 weak Pull-Up resistor on SDA will keep SDA High during the SCL High period, as shown
749 in *Figure 71*.
 - 750 · If the I3C Master wants to abort the Message, then it shall drive SDA Low after the rising
751 edge of SCL, thereby terminating the Message with a Repeated START. The I3C Master
752 then takes control starting with the falling edge of SCL. The Master shall ensure enough
753 delay after SCL rising before driving SDA Low to ensure no contention. In order to
754 achieve this delay, the Master might have to extend the SCL High period.
755 Since the transition of SDA Low when SCL is High is a Repeated START, the Master
756 may begin a new Address (see *Figure 73*), or it may issue a STOP in the next cycle (see
757 *Figure 72*). However in a Mixed Bus the Master should extend the SCL Low period, in
758 order to ensure that any Spike Filters for Legacy I²C Devices properly reset.

- 759 · The Slave shall monitor the SDA on the falling edge of SCL:
- 760 · If SDA is High, then the Slave shall continue with the next data value.
- 761 · If SDA is Low (i.e., if there has been a Repeated START), then the Message has been
- 762 aborted, and the Slave shall not drive SDA after that.

5.1.2.4 Use of Clock Speed to Prevent Legacy I²C Devices From Seeing I3C Traffic

763 In a system, there are three possible I3C Bus Configurations:

- 764 1. **Pure Bus:** Only I3C Devices are present on the Bus.
- 765 2. **Mixed Fast Bus:** Both I3C Devices and Legacy I²C Devices are present on the Bus, such that the
- 766 Legacy I²C Devices are restricted to ones that are generally permissible (i.e., Slave-only, and no
- 767 Slave clock stretching), and that have a true I²C 50 ns Spike Filter on SCL. (I.e., I²C Devices that
- 768 do not “see” the SCL line as High when the High duration is less than 50 ns, across all
- 769 temperatures and processes.)
- 770 3. **Mixed Slow/Limited Bus:** Both I3C Devices and Legacy I²C Devices are present on the Bus,
- 771 such that the Legacy I²C Devices are restricted to ones that are generally permissible (i.e., Slave-
- 772 only, and no Slave clock stretching), but that do not have a true I²C 50 ns Spike Filter on SCL.

773 The Bus designer’s choice of Bus Configuration affects what speed options are available for I3C SDR, as

774 well as what HDR Modes are possible at various clock speeds. **Table 10** shows the possible options for each

775 Bus Configuration.

776 **Table 10 Available Options for Bus Operating Parameters, Per I3C Bus Configuration**

Bus Operating Parameter	Available Options Per Bus Configuration		
	Pure Bus	Mixed Fast Bus	Mixed Slow and Limited Bus
SDR Mode Speed	f _{SCL} (Min) to f _{SCL} (Max)	I ² C messages: Fm or Fm+, I3C messages: SCL High from t _{DIG_H_MIXED} (Min) to t _{DIG_H_MIXED} (Max) with SCL low up to t _{DIG_L} (Max) (see Section 5.1.2.4.1)	Fm or Fm+ only ¹
HDR-DDR Mode	f _{SCL} (Min) to f _{SCL} (Max)	SCL High from t _{DIG_H_MIXED} (Min) to t _{DIG_H_MIXED} (Max) (see Section 5.1.2.4.1)	No
HDR-TSP Mode	f _{SCL} (Min) to f _{SCL} (Max)	Not allowed	No
HDR-TSL Mode	f _{SCL} (Min) to f _{SCL} (Max) (but not needed)	When SCL High, t _{DIG_H_MIXED} (Min) to t _{DIG_H_MIXED} (Max)	No
Note:			
1) May be faster if all Legacy I2C Devices are index 1, per Table 5			

5.1.2.4.1 Use of Duty Cycle to Achieve Lower Effective Speed in a Mixed Fast Bus

777 A pure I3C Bus may simply change the clock speed to any frequency in the allowed speed range, as outlined

778 in **Section 6**. By contrast, a Mixed Fast Bus wanting to clock faster than the slowest Legacy I²C Device needs

779 to take advantage of the Spike Filter, i.e. shall ensure that the SCL High period is shorter than the Spike Filter,

780 in order to prevent the Legacy I²C Devices from seeing the SCL High period (see t_{DIG_H_MIXED} in **Table 75**).

781 As a result, the Legacy I²C Device ‘sees’ the SCL as staying Low the whole period.

782 However, a Mixed Fast Bus I3C Master can change the effective Bus frequency by varying the duty-cycle of

783 SCL. This allows running the data rate slower, for example to accommodate Slaves which need a lower rate

784 as defined by GETMXDS CCC (see **Section 5.1.9.3.18**). It may also be necessary to accommodate Legacy
785 I²C Devices with Spike Filters that need a longer Low period in order to function properly.

786 In this model the SCL High period shall never exceed $t_{DIG_H_MIXED}$, thus staying below the 50ns required by
787 the I²C Spike Filter; however, the Low period is free to be any length permitted by I3C's allowed clock
788 frequency range. For example, depending on the clock generation capability of the I3C Master, the SCL Low
789 period may be a multiple of the High period, or it may be any multiple of some higher frequency clock
790 capable of providing a High period less than or equal to $t_{DIG_H_MIXED}$, but greater than the minimum SCL High
791 period as defined in **Table 73**.

792 **Example 1:** An SCL High period of 40ns, plus a Low period of 280ns, yields a total clock period of 320ns
793 which corresponds to a frequency of 3.125MHz.

794 **Example 2:** In order to ensure that the Legacy I²C Device Spike Filters continue tracking SCL as Low, an
795 SCL High period of 40ns could be used with a Low period of 80ns, yielding a total clock period of 120ns
796 which corresponds to a frequency of 8.3MHz.

797 Such adjustment of the clock Duty Cycle brings three benefits:

- 798 · It remains hidden from Legacy I2C Devices, while still significantly increasing the SDR and HDR-
799 DDR data rate.
- 800 · It ensures a longer Low period, so that Legacy I²C Device Spike Filters continue to track SCL as
801 Low.
- 802 · It is easy for a Master to generate, and has no impact on I3C Slaves (which just react to clock
803 edges).

804 This model works because all I3C Slaves must be able to accommodate 12.5MHz as a clock rate, so the
805 shorter High period will not impact them.

806 Note that this Duty Cycle technique does not resolve issues of long Buses with high line capacitance. This is
807 because the short High period may be insufficient for SDA propagation, even if the Low period is long
808 enough.

5.1.2.5 Master Clock Stalling

809 In SDR Mode the I3C Master may Stall the I3C Bus during the SCL Low period, but only under the specific,
810 transitory conditions described in this Section.

811 Stalling may be necessary for either of two reasons:

- 812 1. The absolute or relative timing of a Message to a specific Slave, or to all Slaves, needs to be
813 carefully controlled. Clock Stalling provides the Master with fine-grained data timing control.
- 814 2. The I3C Master needs to internally synchronize data. This may be due to parts of the Master's
815 system waking up in response to data, to changing state, or to otherwise needing time during a
816 transaction.

817 Note that Stalling impacts Bus performance. For example, it will reduce the Bus capacity and increase the
818 latency of any Devices issuing In-Band Interrupts.

819 To Stall the Bus, the I3C Master shall hold SCL Low while the Bus is in one of the four following conditions,
820 each of which is detailed below:

- 821 1. I3C/I²C Transfer, ACK/NACK Phase
- 822 2. Write Data Transfer, Parity Bit
- 823 3. I3C Read Transfer, Transition Bit
- 824 4. Dynamic Address Assignment, First Bit of Assigned Address

825 The maximum Stall time (SCL Low period) shall be 15 ms; note, this is an absolute maximum. The Master
826 should use the shortest Stall duration possible given current circumstances, as per *Table 11*.

827 **Table 11 Master Clock Stall Times**

Condition	Section	Maximum Stall Time
I3C/I ² C Transfer, ACK/NACK Phase	5.1.2.5.1	100 μs
Write Data Transfer, Parity Bit	5.1.2.5.2	100 μs
I3C Read Transfer, Transition Bit	5.1.2.5.3	100 μs
Dynamic Address Assignment, First Bit of Assigned Address	5.1.2.5.4	15 ms

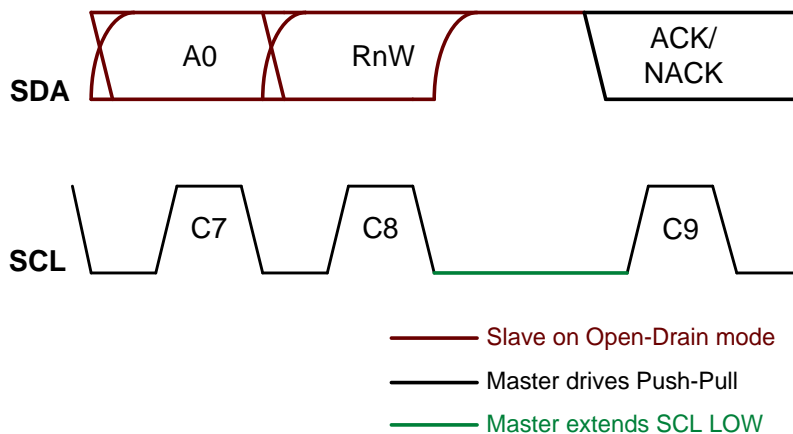
828 In all cases, after Stalling the SCL Low period, the continuation of the clock (i.e., the first SCL rising edge
829 after the extended Low period) shall follow the normal rules for I3C SDR (for I3C SDR Messages), or for
830 I²C (for I²C Messages) including rise time, change in SDA (if any) before SCL rise, next bit, etc. For example,
831 the I3C Master might choose to terminate the Frame after this Stalled bit with a STOP; but in this case the
832 Master is required to observe the requirements for STOP timing, as appropriate.

833 It is recommended to use Master Clock Stalling only when necessary and unavoidable. In all other
834 circumstances the Master should avoid Master Clock Stalling because of its negative impacts on Bus
835 performance. In order to help guide system designers, Data Sheets for I3C Master Devices should include
836 appropriately detailed SCL Stalling parameters.

5.1.2.5.1 I3C/I²C Transfer, ACK/NACK Phase

837 Master Clock Stalling during the ACK/NACK phase of the I3C/I²C transfer (see *Figure 11*) indicates one of
838 the following:

- 839 • The Master can allow the I3C/I²C Slave to prepare to receive data (for write transfers) or transmit
840 data (for read transfers)
- 841 • The Master can Stall SCL during write or read transfers to Legacy I²C Slaves in case of underrun
842 and overflow situations
- 843 • The Master can Stall the clock during the ACK/NACK phase of the incoming In-Band Interrupt
844 (Slave Interrupt Request or Mastership Request), to decide whether to ACK or NACK depending
845 upon the incoming In-Band Interrupt
- 846 • The Master can allow the Slave to respond with data for the directed GET CCC commands if the
847 Slave is not ready with the CCC data to be returned

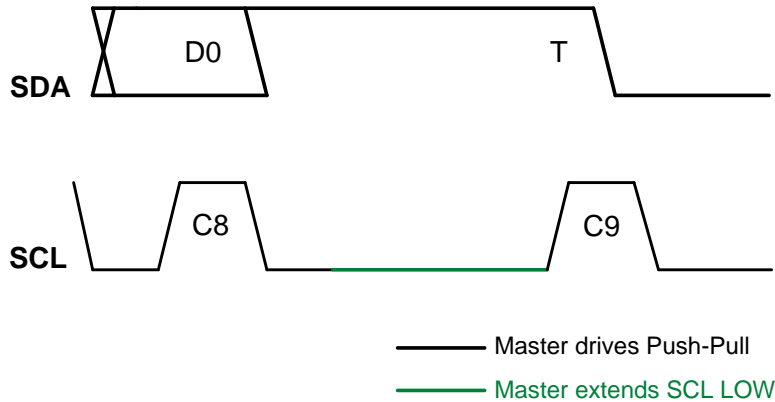


848
849

Figure 11 Master Clock Stalling in ACK Phase

5.1.2.5.2 Write Data Transfer, Parity Bit

850 The Master can Stall SCL between data bytes of an I3C Write Data transfer, by Stalling the clock during the
851 Low period of the Parity Bit, in case of underrun situations. See *Figure 12*.

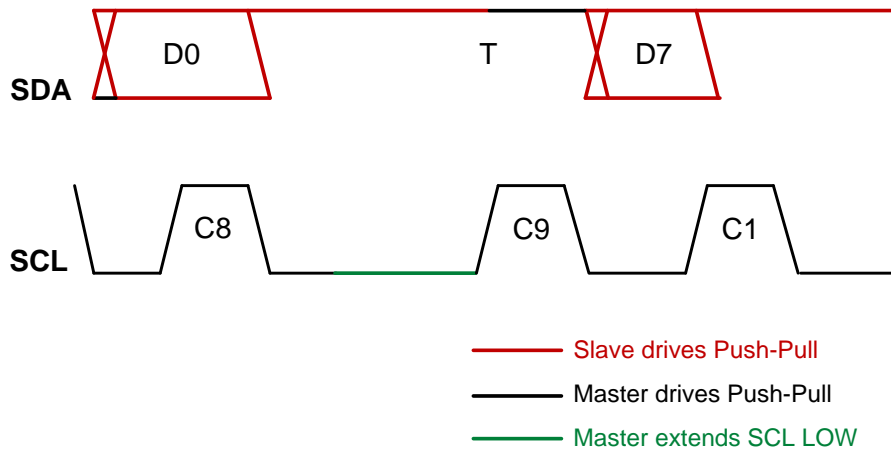


852
853

Figure 12 Master Clock Stalling in Write Parity Bit

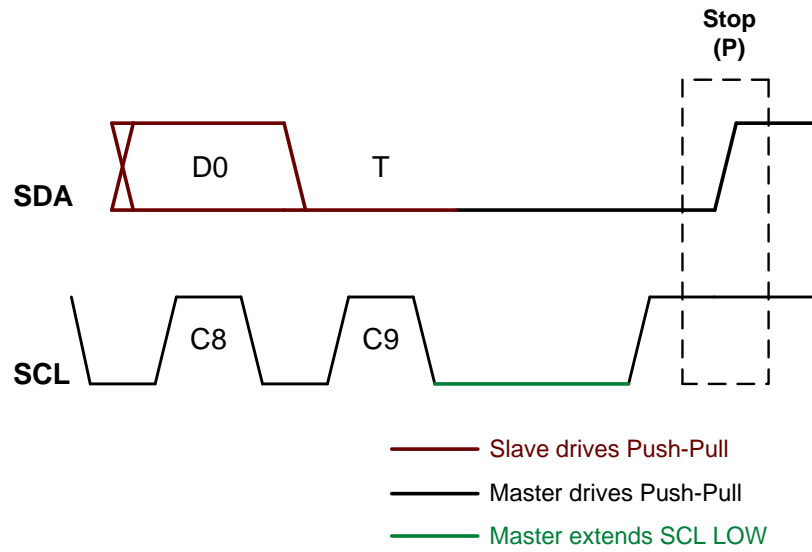
5.1.2.5.3 I3C Read Transfer, Transition Bit

854 The Master can Stall SCL between data bytes of an I3C Read Data transfer, or before terminating, by stalling
855 the clock during the Low period of the Transition Bit, in case of overflow situations. See *Figure 13* through
856 *Figure 17*.



857
858

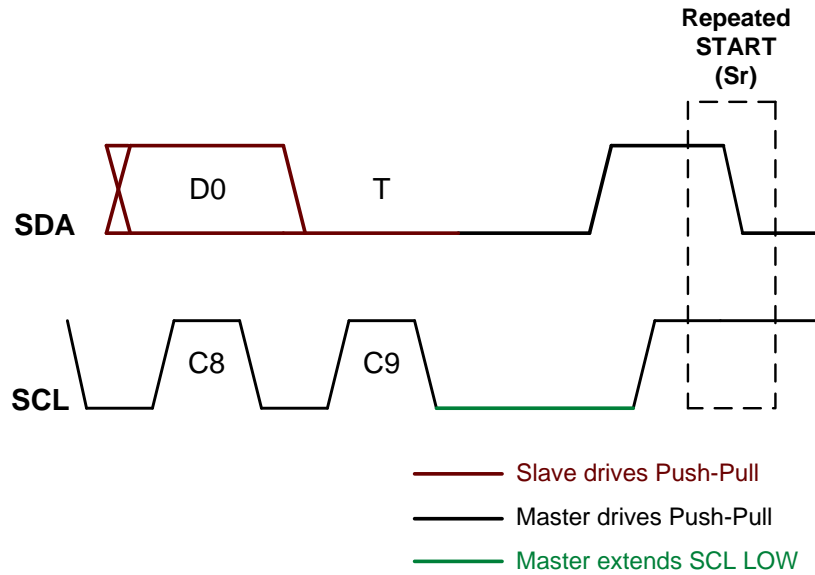
Figure 13 Master Clock Stalling in T-Bit Before Next Read Data



859

860

Figure 14 Master Clock Stalling in T-Bit Before STOP



861

862

Figure 15 Master Clock Stalling in Low T-Bit Before Repeated Start

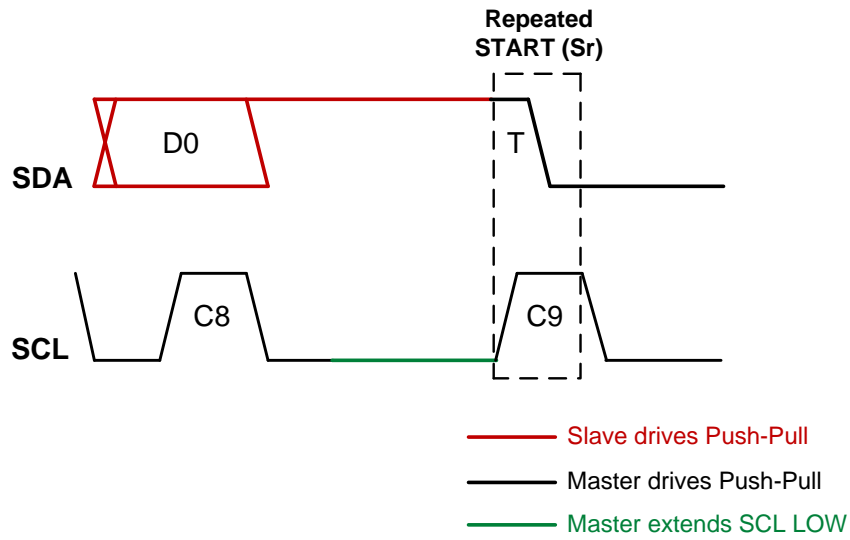


Figure 16 Master Clock Stalling in High T-Bit Before Repeated START

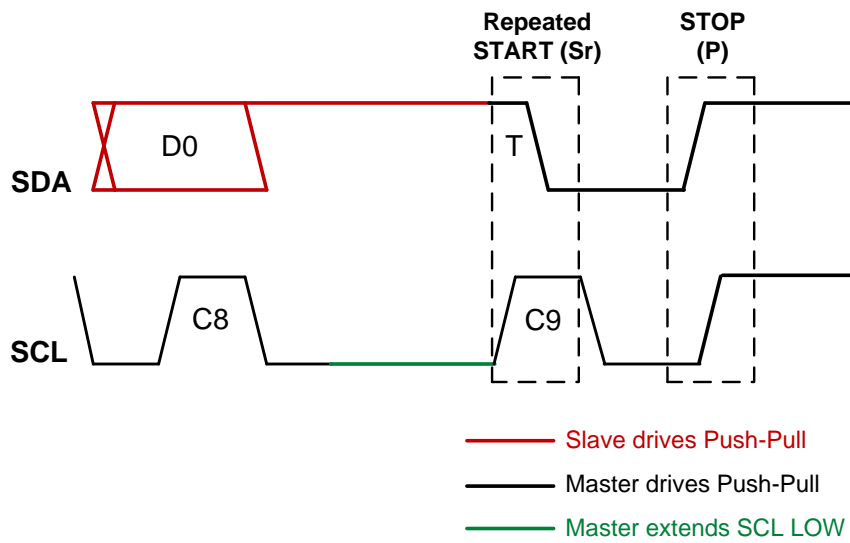
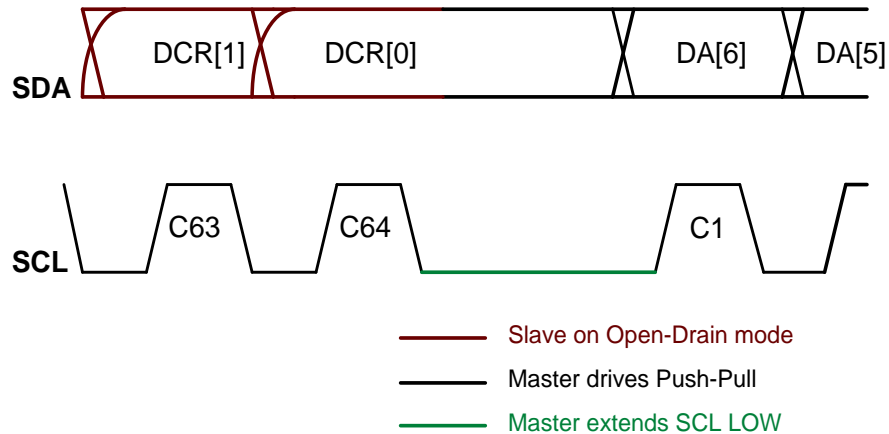


Figure 17 Master Clock Stalling in High T-Bit Before Repeated START and STOP

5.1.2.5.4 Dynamic Address Assignment, First Bit of Assigned Address

867 The Master can stall SCL during the Low period of the first bit of the Assigned Address phase of the Enter
 868 Dynamic Address Assignment CCC command (ENTDAA, see *Section 5.1.9.3.4*), for example to gain time
 869 to assign the Dynamic Address to the Device based on the Slave’s Bus Characteristics Register BCR (see
 870 *Section 5.1.1.2.1*) and Device Characteristics Register DCR (see *Section 5.1.1.2.2*) bytes. See *Figure 18*.



871
872

Figure 18 Master Clock Stalling in Dynamic Address First Bit

5.1.3 Bus Conditions

873 This Specification defines Open Drain Pull-Up and High-Keeper, as well as three distinct conditions in which
874 the I3C Bus shall be considered inactive: Bus Free, Bus Available, and Bus Idle.

5.1.3.1 Open Drain Pull-Up and High-Keeper

875 I3C Master Devices shall provide an active Open Drain class Pull-Up, to be engaged whenever the Bus is in
876 Open Drain mode (with exceptions, detailed below, where a weak Pull-Up may be used).

877 This active Pull-Up shall be implemented either:

- 878 1. As a passive resistance from V_{DD} , or
- 879 2. As a passive resistance from a current source, or
- 880 3. In any other method that both:
 - 881 a. Balances its current sourcing in order to ensure that SDA rises within t_{rDA} (see **Table 73**), and
 - 882 b. Is not so strong as to prevent a Slave with the minimum I_{OL} driver (see **Table 71**) from driving
883 SDA Low within t_{rDA} (see **Table 74**).

884 In addition to the active Open Drain class Pull-Up, a High-Keeper is also required on the Bus. A High-Keeper
885 is generally used for a Master-to-Slave or Slave-to-Master handoff, and for optional termination uses where
886 the Master can signal a termination by driving SDA Low while parked High.

887 The High-Keeper on the Bus shall be strong enough to prevent system leakage (i.e., the sum of the leakage
888 of all Devices on the Bus) from pulling SDA, and sometimes SCL, Low. The High-Keeper on the Bus shall
889 also be weak enough that a Slave with the minimum I_{OL} driver (see **Table 71**) is able to pull SDA, SCL, or
890 both Low within the Minimum t_{DIG_L} period.

891 The Master may provide the High-Keeper on the Bus. If the Master does so, then the Master shall also provide
892 a way to disable its High-Keeper. Reasons to disable a Master-provided High-Keeper might include that the
893 High-Keeper is not strong enough for the present system leakage, or other reasons.

894 A Master-provided High-Keeper may optionally be implemented using a single, common Pull-Up Device
895 capable of supporting both the active Open Drain class Pull-Up function and the weak High-Keeper class
896 Pull-Up function.

897 Whether implemented as a single combined Pull-Up, or as two separate Pull-Ups, the Master should switch
898 SCL and SDA, each independently, between three Pull-Up states as needed, based on Bus state:

- 899 1. No Pull-Up (High-Z)
- 900 2. High-Keeper Pull-Up
- 901 3. Open Drain Pull-Up

902 The Bus shall have High-Keepers on SDA and SCL. When adequate High-Keepers cannot be provided by
903 the Master, then the system designer is required to handle this externally. Reasons why the Master would be
904 unable to provide adequate High-Keepers might include that the Master does not support High-Keepers, that
905 the Master-provided High-Keepers are not strong enough for present needs, or that the Bus is too long to use
906 the Master-provided High-Keepers.

907 The system High-Keepers on SCL and SDA may be implemented as one or more passive resistors tied to
908 V_{DD} , or they may be active Bus-Keeper Devices that turn off when the respective line is pulled below some
909 threshold. The system High-Keepers on SCL and SDA shall be sized so as to balance between system leakage
910 and the requirement that I3C Devices be able to pull the corresponding line Low within the t_{DIG_L} period.

911 **Note:**

912 *The Master may choose to not use the Open Drain class Pull-Up in cases of Open Drain mode where*
913 *the SDA happens to be already High (i.e., is coming into the SCL Falling edge). In that case, the*
914 *Master may choose to rely solely upon the High-Keeper, in order to use less power if and when a*
915 *Slave drives SDA Low.*

5.1.3.2 Bus Free Condition

916 The Bus Free Condition is defined as a period occurring after a STOP and before a START, and with the
917 following duration:

- 918 · **For Pure Bus:** A duration of at least t_{CAS} (see *Table 74*)
- 919 · **For Mixed Bus** (i.e., at least one Legacy I2C is present on the I3C Bus): A duration of at least t_{BUF}
920 (see *Table 73*)

5.1.3.3 Bus Available Condition

921 The Bus Available Condition is defined as a period during which the Bus Free Condition is sustained
922 continuously for a duration of at least t_{AVAIL} (see *Table 74*). A Slave may only issue a START Request (e.g.,
923 for an In-Band Interrupt, or for a Master Handoff Request) after a Bus Available Condition.

5.1.3.4 Bus Idle Condition

924 The I3C Bus Idle Condition is defined in order to help ensure Bus stability during Hot-Join events. The Bus
925 Idle Condition is defined as a period during which the Bus Available Condition (see *Section 5.1.3.3*) is
926 sustained continuously for a duration of at least t_{IDLE} (see *Table 74*).

927 **Note:**

928 *If a Hot-Join Device is powered up onto the I3C Bus at the same time as the Main Master, then the*
929 *Hot-Join Device may pull SDA Low after 1 ms if (1) the Main Master has SCL and SDA pulled up,*
930 *and (2) the Master does not act on the I3C Bus within the same Idle period.*

5.1.3.5 Activity States

931 I3C provides a mechanism for the Master to inform Slaves about expected upcoming levels of activity on the
932 I3C Bus, in order to help the Slaves better manage their internal states. Four Activity State levels from 0
933 through 3 are defined (see *Table 12*).

934

Table 12 Activity States

Activity State	Activity Interval	CCC
0	1 μ second	ENTAS0
1	100 μ second	ENTAS1
2	2 millisecond	ENTAS2
3	50 millisecond	ENTAS3

935 The Activity State number serves as a hint to the Slave, indicating how long it will be before the Master
936 directs Bus activity to that Slave, and the likely latencies to expect in response to the Slave pulling SDA Low
937 (i.e., for the START Request to then generate an In-Band Interrupt Request or a Master Handoff Request).

938 The Master uses CCC commands ENTAS0, ENTAS1, ENTAS2, and ENTAS3 (see *Section 5.1.9.3.2*) to
939 communicate the four expected Bus Activity states to Slaves. Each ENTAS x CCC has both a Broadcast
940 version and a Directed (per-Slave) version. The Master may switch to a different Activity State in any way,
941 and at any time, by issuing the appropriate ENTAS x CCC command.

942 The Slave may use the received Activity State hint to adjust internal settings such as power savings, FIFO
943 trigger levels, timestamp counters and clock rates, and other suitable operating parameters. However Slaves
944 are not required to support the Activity States CCCs (ENTAS x), and as a result could even ignore them
945 completely.

946 The Activity States mechanism is a basis for a general agreement between Master and Slave, i.e., that the
947 Slave may NACK any access occurring sooner than the general time factor. For example: If the Master sends
948 the ENTAS2 CCC, meaning the Master is unlikely to initiate a request sooner than 2 ms from the time the
949 CCC is sent, then a request arriving only 1 ms later could result in a NACK (although it will be ACKED if

950 the request is repeated 50us later, i.e., after the Slave re-awakens). As a result, the Slave shall wake up either
951 upon any Bus activity, or upon matching 7'h7E and its own Dynamic Address.

952 The Activity State CCCs also adjust the maximum value for I3C Bus timing parameter t_{CAS} (Clock after
953 START; see **Table 74**), the maximum amount of time that the Master may take to generate the SCL clock
954 (drive SCL Low) in response to the Slave pulling SDA Low. Note that t_{CAS} is only a worst-case number; it
955 does not indicate whether or not the Master will ACK the In-Band Interrupt Request or Master Request.
956 Further, the selected Activity State does not necessarily indicate the time at which the Master will read
957 additional data from a Slave in response to an In-Band Interrupt; that is a private contract between Master
958 and Slave. A Slave that does not support the ENTASn CCCs shall have a t_{CAS} maximum value of 50
959 milliseconds (the ENTAS3 value).

960 Activity States are not intended as a substitute for a more precise power mode, nor for any other mechanism
961 that might be supported by private contract between Master and Slave. For example, if a Slave has a device
962 power mode setting, then the Master should use that mechanism to put the Slave into the desired state.
963 Likewise, a Slave could provide a FIFO trigger level setting, relating to the amount of time that the Master
964 will have to read the FIFO contents in response to an In-Band Interrupt Request; if so, then the Master should
965 use that setting to match its internal latencies.

5.1.4 Bus Initialization and Dynamic Address Assignment Mode

966 As detailed in this Section, the Main Master is responsible for performing a Dynamic Address Assignment
967 procedure, in order to provide a unique Dynamic Address to each Device connected to the I3C Bus.

968 The Main Master shall provide a Dynamic Address to a Device:

- 969 1. Upon any initialization of the I3C Bus, and
- 970 2. When the Device is connected to an already configured I3C Bus.

971 Once a Device receives a Dynamic Address, that Dynamic Address shall be used in all of that Device's
972 subsequent transactions on the I3C Bus, until and unless the Master changes the Device's Dynamic Address.
973 The only way for the Master to change the Device's Dynamic Address is by using either the RSTDA CCC
974 command (see **Section 5.1.9.3.3**) or the SETNEWDA CCC command (see **Section 5.1.9.3.11**). The Master
975 might choose to change the Device's Dynamic Address due to re-prioritization.

976 The Main Master controls the Dynamic Address Assignment process. This process includes an Address
977 Arbitration procedure similar to I²C's (see **[NXP01]** at **Sections 3.1** and **3.1.8**). The I3C Arbitration procedure
978 differs from I²C by using the values of the 48-bit Provisional ID and the Device's I3C Characteristic Registers
979 (that is, BCR and DCR), concatenated. The Device on the I3C Bus with the lowest concatenated value wins
980 each Arbitration round in turn, and the Main Master assigns a unique Dynamic Address to each winning
981 Device.

5.1.4.1 Device Requirements for Dynamic Address Assignment

5.1.4.1.1 Unique Identifiability

982 In order to support the Dynamic Address Assignment procedure, each I3C Device to be connected to an I3C
983 Bus shall be uniquely identifiable in one of two ways, before starting the procedure.

984 **Either:**

- 985 1. The Device may have a Static Address, in which case the Master may use that Static Address
986 (presuming it is known to the Master).

987 For example, an Address similar to what I²C specifies [*NXP01*].

988 **Or:**

- 989 2. The Device shall in all cases have a 48-bit Provisional ID. The Master shall use this 48-bit
990 Provisional ID, unless the Device has a Static Address and the Master uses the Static Address.

991 The 48-bit Provisional ID is composed of three parts:

- 992 1. **Bits [47:33]:** MIPI Manufacturer ID [*MIPI01*] (15 bits)

993 *Note:* The Most Significant Bit of the MIPI Manufacturer ID is discarded, i.e.
994 only the 15 Least Significant Bits are used.

- 995 2. **Bit [32]:** Provisional ID Type Selector (One bit, 1'b1: Random Value, 1'b0: Vendor Fixed
996 Value)

- 997 3. **Bits [31:0]:** 32 bits containing either a Vendor Fixed Value or a Random Value, depending on
998 the value of Bit [32]:

999 **If the value of Bit [32] is 1'b0: Vendor Fixed Value:**

- 1000 • **Bits [31:16]: Part ID:** The meaning of this 16-bit field is left to the Device vendor to
1001 define.
- 1002 • **Bits [15:12]: Instance ID:** The value in this 4-bit field should identify the individual
1003 Device, using a method selected by the system designer. For example: straps, fuses, non-
1004 volatile memory, or another appropriate method.
- 1005 • **Bits [11:0]:** The meaning of this 12-bit field is left for definition with additional meaning.
1006 For example: deeper Device Characteristics, which could optionally include Device
1007 Characteristic Register values.

1008 **If the value of Bit [32] is 1'b1: Random Value:**

- 1009 • Bits [31:0]: 32-bit value randomly generated by the Device. This value can be queried via
1010 General Test Mode, using the Command Code **Enter Test Mode (ENTTM)** (see
1011 *Section 5.1.9.3.8*).

1012 **Note:**

1013 *Under Vendor Test Mode, the Device may provide a fully random or pseudo-random 48-bit*
1014 *Provisional ID (see Section 5.1.9.3.8).*

5.1.4.2 Bus Initialization Sequence with Dynamic Address Assignment

1015 Bus Initialization and Dynamic Address Assignment shall be executed according to the following sequence.
1016 See also *Figure 90 (Dynamic Address Assignment FSM)*.

1017 The Dynamic Address Assignment process shall be performed in Open Drain mode, except that the Repeated
1018 START and 7'h7E/R may be either Open Drain or Push-Pull. For Open Drain the Main Master shall drive
1019 the SCL line with clocks at the appropriate Open Drain speed for the Devices present on the I3C Bus. For
1020 Repeated START, the Main Master may optionally choose to actively drive the SDA line High, but only after
1021 the Slave has safely released the SDA line.

- 1022 1. The Main Master shall begin in an appropriately configured state, and shall have, either in its own
1023 non-volatile memory or as a result of having received it from the Application Host, the following
1024 data:
- 1025 a. The number of I3C compliant Devices that need to receive a Dynamic Address,
1026 b. The data for any I3C Devices resident on the I3C Bus that already have Static Addresses, and
1027 c. The data for any Legacy I²C Devices resident on the I3C Bus.
- 1028 2. The Main Master shall assign Dynamic Addresses to any I3C Devices with a known Static
1029 Address, using the Command Code **Set Dynamic Address from Static Address (SETDASA)**
1030 (see *Section 5.1.9.3.10*).
- 1031 3. The Main Master shall send the Broadcast Command Code **Enter Dynamic Address Assignment**
1032 (**ENTDAA**) (see *Section 5.1.9.3.4*). There is no data associated with this Command Code.
- 1033 4. The Main Master shall send a Repeated START, and the I3C Broadcast Address 7'h7E with RnW
1034 bit High (i.e. Read). Every I3C Device on the I3C Bus that does not yet have an assigned Dynamic
1035 Address, and that is not a Hot-Join Device, shall acknowledge the I3C Broadcast Address. (I.e.,
1036 Hot-Join Devices that do not yet have an assigned Dynamic Address shall not acknowledge the
1037 I3C Broadcast Address in this step.)

1038 At least one I3C Device on the I3C Bus will acknowledge the I3C Broadcast address in this step.

1039 **Note:**

1040 *This use of the I3C Broadcast Address (7'h7E) with RnW bit High (i.e. Read) is specific to Dynamic*
1041 *Address Assignment Mode. It is also acceptable per the PC Specification [NXP01].*

- 1042 5. The Main Master shall drive only the SCL line. The Main Master shall release the SDA line to a
1043 High-Z state, allowing SDA to go to High level via the Bus Pull-Up resistor.
- 1044 6. Every I3C Device that responds to the I3C Broadcast Address sent in Step 4 shall drive the SDA
1045 line with its own 48-bit Provisional ID (using Big Endian bit order), until it loses Dynamic
1046 Address Arbitration.
- 1047 a. The 48-bit Provisional ID shall be transferred continuously, starting with the most significant
1048 bit (bit [47]), with no delimitation or ACK/NACK pulse.
- 1049 b. The Hot-Join Devices shall participate in the Dynamic Address Assignment procedure only
1050 after requesting it via an In-Band Interrupt using the Reserved Slave Address of 7'b0000_010
1051 and RnW bit Low (i.e. Write).
- 1052 7. The Main Master shall continue to drive the SCL line with the same clock, while still releasing the
1053 SDA line. The I3C Device that did not yet lose the Arbitration shall then transfer its Bus
1054 Characteristics Register (BCR) and Device Characteristic Register(s) (DCR) per *Section 5.1.1.2.2*,
1055 until it eventually loses Dynamic Address Arbitration. See also *Section 5.1.4.3*.
- 1056 8. The Device whose concatenated Provisional ID, BCR, and DCR has the lowest value will win the
1057 Arbitration round, due to the nature of Arbitration.

1058 **Note:**

1059 *It is possible for multiple Devices to have the same concatenated value, although this is highly*
1060 *improbable. See **Section 5.1.4.3**.*

- 1061 9. The Main Master shall transfer a 7-bit wide Dynamic Address for the winning Device, in Open Drain
1062 Mode. This Dynamic Address shall incorporate the priority level that the Main Master assigns to the
1063 Device, per *Section 5.1.6.2*. The Arbitration-winning Device shall acknowledge the assigned
1064 Dynamic Address. This Dynamic Address transfer procedure shall have the following steps:
- 1065 · The Main Master shall drive the 7-bit Dynamic Address, followed by a parity bit (PAR), which
1066 is calculated as odd parity. Odd parity is the inverse of the XOR of the 7 bits. Therefore
1067 $\sim\text{XOR}(\text{dynamic_address}[7:1])$ is placed in position 0.
 - 1068 · If the parity is valid, then the Slave shall acknowledge receipt of the Dynamic Address on the
1069 next SCL clock. If the parity is invalid, then the Slave shall passively NACK on the next SCL.

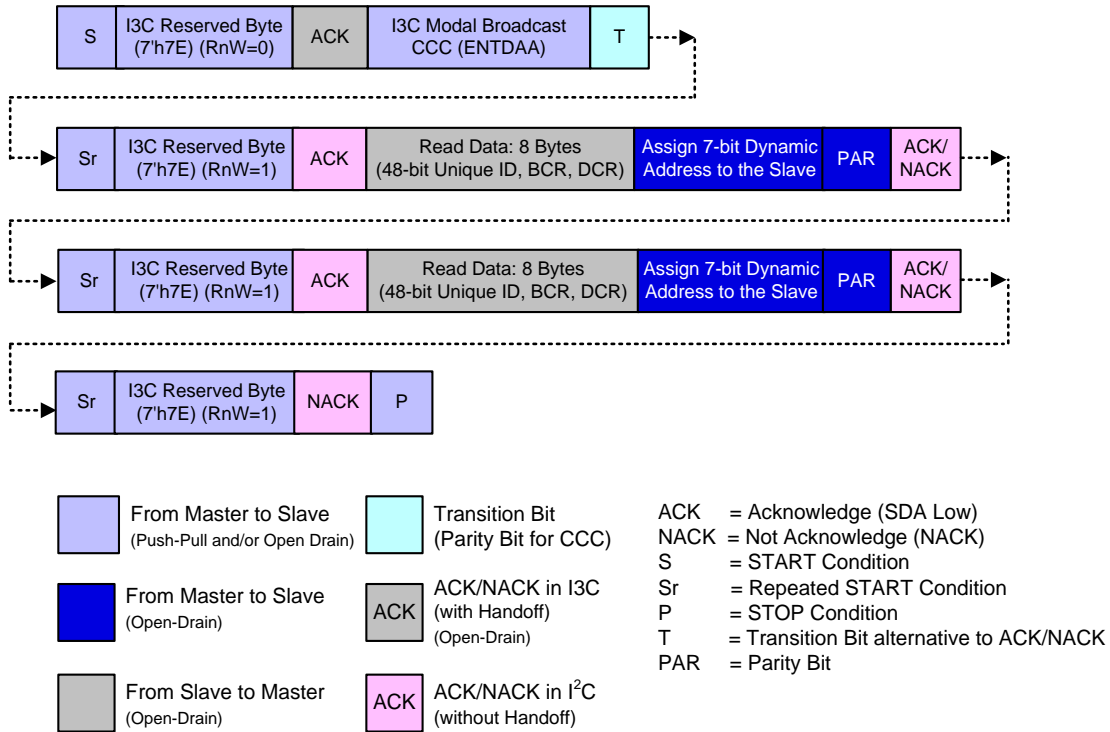


Figure 19 Dynamic Address Assignment Transaction

1070

1071

1072

1073

1074

1075

1076

1077

1078

1079

1080

1081

1082

1083

1084

1085

1086

1087

1088

1089

1090

1091

1092

1093

1094

1095

10. The Main Master shall repeat this procedure, jumping back to step 4 until there is no ACK from any Device present on the I3C Bus.

11. This Dynamic Address Assignment procedure shall be ended by the Main Master issuing a STOP.

Regarding this Dynamic Address Assignment procedure, note that:

- The Main Master is able to end the Dynamic Address Assignment procedure at any time, even if some of the pre-established I3C Devices have not yet received their Dynamic Addresses.
- The Dynamic Address Assignment procedure can be started again any time the I3C Bus is in Bus Available Condition, using the Command Code **Enter Dynamic Address Assignment (ENTDAA)** (see **Section 5.1.9.3.4**).
- If a given Slave does not acknowledge its assigned Dynamic Address, then the procedure requires the Main Master to continue from step 4. The Slave will then participate in the Address Arbitration using the same 48-bit Provisional ID, and as a result the Slave will win the Arbitration round. If the Slave does not ACK the Dynamic Address a second time, then the Main Master shall exit the Dynamic Address Assignment procedure and execute an error management procedure provided by the I3C Bus designer.
- Any Secondary Masters wishing to receive the Dynamic Addresses assigned to all Devices present on the I3C Bus can do one, or both, of the following:
 - Follow the Dynamic Address Allocation procedure. A Secondary Master that follows the same procedure will have received all the same data.
 - Use the Command Code **Define List of Slaves (DEFSLVS)** (see **Section 5.1.9.3.7**) to acquire (eventually selectively) the Addresses and Characteristics of all Devices on the I3C Bus from the Main Master.

After the Dynamic Address Assignment process is complete, the Main Master knows which Device on the I3C Bus is the Secondary Master. The Main Master then addresses the Secondary Masters with the Command

1096 Code **Define List of Slaves (DEFSLVS)** (see *Section 5.1.9.3.7*) and transfers the data for any and all Legacy
1097 I²C Devices on the I3C Bus. In order to keep the I3C Bus under control, this could all be done within the
1098 same transaction (i.e. the Main Master continues the communication by issuing a Repeated START each
1099 time, with no intervening STOP).

5.1.4.3 Provisional ID Collision Detection and Correction

1100 In the Dynamic Address Assignment procedure described in *Section 5.1.4.2*, multiple I3C compliant Devices
1101 will receive the same Dynamic Address if they provide the Main Master with both identical 48-bit Provisional
1102 IDs, and identical Device Characteristic Register values. Although the probability of such a coincidence is
1103 quite small the potential does exist, and the I3C Bus will not operate properly under such conditions (at least
1104 the Instance IDs must be different, see *Section 5.1.4.1.1*). As a result the Main Master must test for this
1105 collision condition, and resolve it if necessary, before the I3C Devices present on the I3C Bus can all be
1106 safely used together.

1107 The I3C Main Master knows the number of Devices resident on the I3C Bus requiring Dynamic Address
1108 Assignment (per procedure step 1.a. in *Section 5.1.4.2*), which enables detection of collision errors. At
1109 completion of the Dynamic Address Assignment procedure the Main Master shall compare this expected
1110 number to the final count of Static Addresses and Dynamic Addresses that were actually assigned. If fewer
1111 Dynamic Addresses were assigned than expected, then multiple Devices must have received the same
1112 Dynamic Address, i.e. a collision has occurred.

1113 If this collision condition is detected, then the Main Master shall resolve it using either of the following
1114 methods:

- 1115 • The Main Master directs a Reset Dynamic Address request to the Dynamic Address most likely to
1116 have been duplicated, using the Command Code **Reset Dynamic Address Assignment**
1117 (**RSTDAA**) (see *Section 5.1.9.3.3*).
- 1118 • Alternatively, the Main Master resets the Dynamic Address Assignment process by using the
1119 Broadcast Command Code **Reset Dynamic Address Assignment (RSTDAA)** (per
1120 *Section 5.1.9.3.3*), and then proceeding from step 3 of the procedure in *Section 5.1.4.2* from step 8,
1121 until the sooner of either:
 - 1122 a. The expected number of I3C Devices is discovered, or
 - 1123 b. A set maximum number of attempts fail. If this limit is reached, then a system error message
1124 shall be sent to the Host. To avoid freezing the entire system, a limit of three (3) attempts is
1125 recommended.

5.1.5 Hot-Join Mechanism

1126 The I3C protocol supports a Hot-Join mechanism, to allow Slaves to join the I3C Bus after it is already
1127 configured.

1128 **Note:**

1129 *Hot-Join does not allow Slaves to join the I3C Bus before the I3C Bus has been configured.*

1130 Hot-Join may be used for:

- 1131 • I3C Devices mounted on the same board, but de-powered until needed. Such Devices shall not
1132 violate electrical limits for Slaves when depowered (or while transitioning) as defined in **Section 6**,
1133 including capacitive load.
- 1134 • I3C Devices mounted on a module/board that is physically inserted after the I3C Bus has already
1135 been configured. This specification does not attempt to address how that physical insertion is
1136 handled, however such insertion shall not disrupt the SCL and SDA lines, including respecting all
1137 electrical limits defined in **Section 6**.

1138 Hot-Joining Slaves may be any valid Slave type, including Secondary Master. After a Hot-Join, the Current
1139 Master shall use the **DEFSLVS** CCC as described in **Section 5.1.9.3.7**. To ensure that any Secondary Masters
1140 are aware of all of the available Slaves, the Master shall immediately notify the I3C Bus if it discovers that
1141 any previously joined Slaves are no longer present on the Bus (e.g. due either to non-response, or to
1142 mechanisms outside of the Bus).

1143 Since the Master is not inherently aware when new Slaves join the I3C Bus, the Hot-Join mechanism provides
1144 the following procedure for informing the Master about new Slaves:

- 1145 1. The Slave shall wait for at least t_{IDLE} of BUS Available Condition (see **Table 74**).
- 1146 2. After the Slave determines that the Bus is Idle it may either issue a START, or else wait for a
1147 START. (To issue a START, pull the SDA line Low and hold it Low until the SCL line goes Low.)

1148 **Note:**

1149 *The Hot-Join Device may power-up at the same time as the Main Master (e.g. may be connected to
1150 the I3C Bus when power is applied to the system). In that case, the Main Master might pull SDA Low
1151 even before the I3C Bus has been started, assuming that SCL and SDA are being pulled up. If a
1152 Main Master needs 1 ms or more in order to start acting on the I3C Bus, then that Main Master shall
1153 be able to accommodate the situation of SDA being held Low when it is ready to initialize the I3C
1154 Bus, or it may instead delay pulling SDA High, and/or pulling SCL High, until it is ready.*

1155 Hot-Join capable Devices should implement some method that allows the system designer to
1156 prevent the Device from attempting to Hot-Join when the Device is not being used as a Hot-Join
1157 Device. For example: use of NVMEM, or else a pin strap could be used when the Device is
1158 soldered onto the board and powered with the rest of the I3C Bus.

- 1159 3. The Slave issues the Reserved Slave Address of 7'b0000_010 as an IBI, using W (write) after the
1160 START. This requests a Dynamic Address Assignment process.
- 1161 4. The Current Master shall perform one of 3 actions:
 - 1162 a. NACK the request. The Slave shall try again at the next START.
 - 1163 b. ACK the request, but issue a Broadcast CCC to disable Hot-Join by setting the DISHJ bit in
1164 the Command Code **Enable/Disable Slave Events Command (ENEC/DISEC)** (see
1165 **Section 5.1.9.3.1**). If another Device attempts to Hot-Join before the Master is ready to assign
1166 Dynamic Addresses to the Slaves, then the Master might need to repeat this procedure.
 - 1167 c. ACK the request and then issue a Broadcast Command Code **Enter Dynamic Address
1168 Assignment (ENTDAA)** (see **Section 5.1.9.3.4**) to start the Dynamic Address Assignment
1169 process. Since only Slaves with no Dynamic Address shall participate, this allows the newly
1170 joined Slave to receive its Dynamic Address.
- 1171 5. If an **Enter Dynamic Address Assignment (ENTDAA)** Command Code is issued as described in
1172 **Section 5.1.9.3.4**, then the Slave shall transmit its 48-bit Address as per the normal mechanism.

1173 Hot-Join Slaves required to receive the same assigned Dynamic Address every time they join can
1174 use a fixed-value ID, per *Section 5.1.4.1.1*.

1175 If a Slave drops off the I3C Bus due to being detached or depowered, then the Master might not be aware of
1176 this. The Master may determine this condition by repeated attempts to contact the Slave using a safe (shall
1177 respond) Command Code such as **Get Device Status (GETSTATUS)** (see *Section 5.1.9.3.15*). If the Master
1178 determines that the Slave is no longer part of the I3C Bus, then it shall either recycle that Slave's Dynamic
1179 Address, or else reserve that Slave's Dynamic Address in case that Slave later re-joins the Bus.

5.1.5.1 Failsafe Device Pads

1180 Hot-Join Devices shall be Failsafe, unless there is a protection method outside of the pad. Failsafe means that
1181 when the SCL and SDA pads are unpowered but wire connected to the I3C Bus, they must not draw extra
1182 leakage current from the Bus.

1183 The Device SCL and SDA pads shall be designed so as to avoid drawing current from the system SCL and
1184 SDA lines, both when they are being held High and when they are being held Low. Such leakage may be
1185 avoided by using special connectors, or other isolation methods (e.g., physical connection order in a plug).
1186 This excess current should be avoided whether due to diode effect, rail tie for ESD, or clamps. The leakage
1187 current variation between unpowered and powered shall not exceed the normal variation range of an active
1188 pad, I_i , as specified in *Section 6, Table 71*.

5.1.6 In-Band Interrupt

1189 This Section specifies I3C's priority-based In-Band Interrupt mechanism.

5.1.6.1 Priority Level

1190 In I3C, Priority Level controls the order in which Slaves' In-Band Interrupt requests and Master requests are
 1191 processed. The Priority Level of each Slave in an I3C Bus instantiation is encoded in its Slave Address, with
 1192 lower Addresses having higher Priority. That is, Slaves with lower value Addresses and higher Priority Levels
 1193 have their In-Band Interrupts and Master requests processed sooner than Slaves with higher value Addresses
 1194 and lower Priority Levels. This Priority Level ordering is a natural outcome of the I3C Address Arbitration
 1195 behavior specified at *Section 5.1.2.2.1*, where Address bits with value 0 prevail bits with value 1.

1196 As a result, during each Dynamic Address Assignment operation (see *Section 5.1.4*) the Main Master should
 1197 assign lower Addresses to Slaves for higher Priority In-Band Interrupt requests.

5.1.6.2 I3C Slave Interrupt Request

1198 In order to request an interrupt, an I3C Slave shall emit its Address into the arbitrated Address header
 1199 following a START (but not following a Repeated START). If no START is forthcoming within the Bus
 1200 Available Condition, then the I3C Slave may issue a START Request by pulling the SDA line Low. In
 1201 response, the Main Master shall set the SCL clock line Low, completing the START Condition. The Main
 1202 Master shall process interrupt requests in Priority Level order, per *Section 5.1.6.1*. The Slave shall then drive
 1203 the SDA with its own Address, followed by an RnW bit of 1.

1204 At that point, the Current Master shall do one of the following three things:

- 1205 1. Accept the IBI by providing the ACK bit. The actions available to the Current Master depend upon
 1206 the value of the Slave's BCR [2] bit (in the Slave's BCR register):
 - 1207 a. If the I3C Slave's BCR [2] bit is set to 1, then, per *Section 5.1.1.2.1*, the Current Master shall
 1208 read the Mandatory Data Byte that follows the accepted IBI request at any "read" clock speed
 1209 allowable by the Slave. This operation is similar to a "read" from the Slave and all the related
 1210 rules apply. Note that the Current Master cannot avoid receiving the Mandatory Data Byte,
 1211 since it is transmitted in Push-Pull mode.

1212 After reading the Mandatory Data Byte, the Current Master may take any other valid I3C
 1213 action. For example, the Current Master could issue a STOP, or issue a Repeated START,
 1214 or it could continue reading additional Data Bytes from the Slave (if, for example, a
 1215 private contract between the two Devices has been established in advance).

1216 One conceptual time diagram of this sequence is shown in *Figure 20* below:

Open Drain	Open Drain	Open Drain	Hand Off	Push-Pull	Drive High or Low, and then High-Z	Push-Pull
S	Slave_addr_as_IBI/R	Master_ACK	SCL High	Slave_byte	T	Sr

1217 **Figure 20 IBI Sequence with Mandatory Data Byte**

- 1218 b. If the I3C Slave's BCR [2] bit is set to 0, then the Current Master may take any other valid
 1219 I3C action. For example, the Current Master could issue a STOP, or issue a Repeated START,
 1220 or it could continue reading additional Data Bytes from the Slave at any allowable "read"
 1221 clock speed (if, for example, a private contract between the two Devices has been established
 1222 in advance).
- 1223 2. Refuse the IBI without disabling interrupts. To do this, the Current Master simply passively
 1224 NACKs to deny the IBI. (The Current Master knows that the Slave should try to interrupt again on
 1225 the next START, which will cause the Current Master to issue a Repeated START.)

- 1226 3. Refuse the IBI and disable interrupts. To do this, the Current Master NACKs to deny the IBI, then
1227 sends a Repeated START, and finally sets the DISINT bit in the Command Code **Disable Slave**
1228 **Events Command (DISEC)** to the interrupting Slave as described in *Section 5.1.9.3.1*. (The
1229 Current Master can set the ENINT bit in the Command Code **Enable Slave Events Command**
1230 **(DISEC)** at a later time.)

1231 In cases where the Main Master anticipates that the I3C Bus might be needed for a higher Priority Level
1232 transaction before the newly requested transaction would complete, the Master could either deny the access
1233 (by not acknowledging the request), or else drive the higher-priority Device Address instead of the Address
1234 of the Device sending the IBI request. The Main Master would then hold the SCL line Low until the expected
1235 higher-priority transaction begins.

1236 Bus contention occurs during Address evaluation. As a result, if multiple I3C Devices simultaneously attempt
1237 to win the Bus, then all but one will lose the Arbitration. These Devices will have the opportunity, but are not
1238 required, to repeat the attempt upon the next Bus Available Condition. Note that Slave Devices have the
1239 option to choose to not try again.

5.1.6.3 I3C Secondary Master Requests to be Current Master

1240 When the I3C Secondary Master requests to be a Current Master:

- 1241 1. To perform the request, the I3C Secondary Master shall wait for either a START (not a Repeated
1242 START), or a Bus Available Condition and issue its own START.
- 1243 2. After a START, the I3C Secondary Master shall issue its own Dynamic Address on the I3C Bus,
1244 followed by the RnW bit of 0 to request to become Current Master.
- 1245 3. If the I3C Secondary Master wins the Arbitration by having the lower Dynamic Address, and if the
1246 Current Master is currently willing to relinquish Mastership to the requester, then the Current
1247 Master shall respond with ACK.
- 1248 4. After the ACK, the Current Master may issue one or more commands, and shall then issue a
1249 GETACCMST CCC (see *Section 5.1.9.3.16*) followed by a STOP, whereupon it releases control
1250 of the SCL line, and therefore also releases Master control of the I3C Bus.
- 1251 5. Following the STOP and Bus Available Condition, the I3C Secondary Master assumes the role of
1252 the Current Master and takes control of the I3C Bus.

1253 *Figure 74* presents a simplified and generalized picture of the Master to Master Handoff procedure. While
1254 there are many possible variations to this process, the most significant states are as follows:

- 1255 1. At the end of data transfer, the Current Master (indicated by ‘CM’ in the Figure) controls SCL and
1256 SDA, and the New Master (‘NM’ in the Figure) has SCL and SDA in High-Z state.
- 1257 2. The Current Master provides the rising edge of SCL, while keeping SDA Low
- 1258 3. After t_{SU_SRO} elapses, the Current Master drives SDA High using either an active drive, or the
1259 Open-Drain class Pull-Up. However, once SDA is High the Open Drain class Pull-Up shall be
1260 used.
- 1261 4. After the New Master determines that SDA is High, and after both its Clock to Data Turnaround
1262 time and the time of flight elapse, it then takes two actions:
 - 1263 a. The New Master Actively drives SCL High, overlapping with the Current Master’s active
1264 drive; and
 - 1265 b. The New Master enables its Open-Drain class Pull-Up, in parallel with the Current Master’s
1266 Open-Drain class Pull-Up

1267 Neither of these actions conflicts with the Current Master.

1268 Although t_{SCO} generally applies to a Slave Device, in this step it applies to the New Master
1269 because prior to the Bus transfer the New Master performs in the Slave role.

1270 In the Figure, the t_{SCO} is shown starting as if the CM has used the Open-Drain drive of SDA High;
 1271 this depicts the worst condition for this stage of the handoff, showing the latest possible moment
 1272 when the NM starts overlapping the line controls with the CM.

- 1273 5. After a time delay of $t_{MMoverlap}$, the Current Master:
 1274 a. Releases SCL to High-Z; and
 1275 b. Disables its Open-Drain class Pull-Up on SDA, and sets SDA to High-Z

1276 In the Figure, $t_{MMoverlap}$ is shown in the worst case: on the Open-Drain drive of SDA, and where
 1277 the CM starts counting it from the lower side of the SDA rising edge. The CM must control the
 1278 lines until the NM could safely take over, and that is certain after $t_{DIG_OD_L\ Min}$. As a result,
 1279 $t_{MMoverlap}$ shall be greater than or equal to $t_{DIG_OD_L\ Min}$.

- 1280 6. After t_{MMlock} (the time interval during which the New Master shall not drive SDA Low), the New
 1281 Master could actively drive SDA Low, producing a START condition.

1282 The Slaves may also drive SDA Low at this point. This is allowed because SDA is held by an
 1283 Open-Drain class Pull-Up. The t_{MMlock} period timing parameter is similar to I3C's t_{AVAL} and I²C's
 1284 t_{BUF} and, as such, requires different values for Pure Bus and for Mixed Bus, respectively.

1285 In the Figure, t_{MMlock} is shown as referred to the SDA rising edge in Open-Drain driving mode,
 1286 similar to I3C's t_{AVAL} and I²C's t_{BUF} .

- 1287 7. After t_{CAS} expires, the New Master may drive SCL Low, producing the first SCL falling edge.

1288 Following this SCL falling edge, the New Master shall actively drive SCL, while also driving SDA in Open-
 1289 Drain mode with the arbitrable address.

1290 **Note:**

1291 *The way the I3C Secondary Master indicates that it is a Secondary Master is by properly returning*
 1292 *the value 2'b01 for Device Role in its BCR (see **Section 5.1.1.2.1**).*

1293 *The I3C Secondary Master records the Dynamic Addresses of all Devices on the I3C Bus. The*
 1294 *preferred method is to record the Address from the Command Code **Define List of Slaves***
 1295 *(**DEFSLVS**) (see **Section 5.1.9.3.7**).*

1296 *The Master determines that this is a Current Master request by returning the value 2'b01 for Device*
 1297 *Role in its BCR (see **Section 5.1.1.2.1**).*

5.1.6.4 I3C Main Master Initiating a Transaction

1298 When initiating an I3C transaction, an I3C Current Master shall perform Dynamic Address Arbitration during
 1299 the Address call. Any other I3C Device attempting to interrupt shall win the Arbitration. Interrupting Devices
 1300 with lower-priority level shall wait for the next Bus Available Condition.

1301 **Note:**

1302 *In order to allow the lower-priority Slaves to perform an In-Band Interrupt, Masters may do the*
 1303 *following:*

- 1304 1) *Wait more than the minimum "Bus Available" period before starting a new communication.*
 1305 2) *Transmit the reserved I3C Broadcast Address, followed by a Repeated START and normal*
 1306 *Messages. (The reserved I3C Broadcast Address has lower priority than any interrupts.)*

5.1.7 Secondary Master Functions

1307 Once granted control of the Bus, the Secondary Master maintains control until another Master is granted Bus
1308 control. After the Secondary Master transitions to the Current Master role it could encounter Bus management
1309 activities besides the data transfers that it itself initiates. Some examples are the In-Band Interrupt, or the
1310 Hot-Join request. One optional possibility, shown at *Section 5.1.7.2*, is that the Secondary Master performs
1311 the Current Master's actions with the full capabilities of the Main Master. Another optional possibility is that
1312 the Secondary Master, while serving in the Current Master role, could defer some actions to a more capable
1313 Master, as described in *Section 5.1.7.3*.

1314 A Secondary Master shall support the following scenarios according to its capabilities.

5.1.7.1 Hardware and Software Requirements

1315 The Secondary Master capable of performing the Main Master tasks shall have the following minimum
1316 hardware and software capabilities:

- 1317 1. Memory for:
 - 1318 a. All Bus Devices' capabilities and functions at system level
 - 1319 b. Retaining the Dynamic Addresses of all I3C Bus Devices
 - 1320 c. Retaining the data transfer protocol that the I3C Bus Devices are capable of
- 1321 2. Link requirements for Slaves that are intended to transmit In-Band Interrupt requests (IBI), e.g. the
1322 clock speed and the maximum length of data transfer
- 1323 3. Data transfer protocol capability needed for communicating to all I3C Bus Devices

5.1.7.2 Bus Management Procedures

1324 The Secondary Master capable of performing the Main Master tasks shall perform the following minimum
1325 Bus Management procedures:

- 1326 1. During Bus initialization, the Secondary Master shall obtain Characteristics of Devices on the Bus
1327 by either using the Command Code **Define List of Slaves (DEFSLVS)** (see *Section 5.1.9.3.7*) sent
1328 by the Main Master, or by monitoring the Bus during Dynamic Address assignment.
- 1329 2. Perform the Dynamic Address Assignment for Hot-Join Devices
 - 1330 a. The Secondary Master needs to know all Bus Devices' functionality at system level, and such
1331 in order to be able to assign the correct priority level to the Hot-Join Device
 - 1332 b. The Current Master shall issue the Command Code **Define List of Slaves (DEFSLVS)** (see
1333 *Section 5.1.9.3.7*) to ensure that other Masters have the same knowledge
- 1334 3. Manage the In-Band Interrupt procedure
1335 The Secondary Master needs to know the meaning of the interrupt from the Slave, and needs to
1336 service the interrupt appropriately.
- 1337 4. Procedure for requesting transfer of Bus control to another Master, including the Main Master
1338 The Secondary Master uses the Command Code **Get Accept Mastership (GETACCMST)** (see
1339 *Section 5.1.9.3.16*).

5.1.7.3 Reduced Functionality Secondary Masters

1340 Secondary Masters are not required to implement and be able to perform all the tasks required of the Main
1341 Master. The Secondary Master may defer or transfer the Bus control to a Master when needed; see
1342 *Section 5.1.7.1* and *Section 5.1.7.2*.

1343 A Secondary Master with reduced functionality shall have the following minimum capabilities:

- 1344 1. Memory sufficient for retaining Device Addresses and Characteristics of any Slaves it supports
- 1345 2. Memory sufficient for retaining the Device Address and Characteristics of the Master to which it
1346 will return Bus control

5.1.7.4 In-Band Interrupt Handling

1347 The Secondary Master can either service the In-Band Interrupt request as described in *Section 5.1.6*, or else
1348 defer In-Band Interrupt handling to a capable Master by setting the DISINT bit in the Command Code
1349 **Disable Slave Events Command (DISEC)** to the interrupting Slave (see *Section 5.1.9.3.1*).

5.1.7.5 Hot-Join Management

1350 The Secondary Master can either manage the Hot-Join event as described in *Section 5.1.5*, or else defer
1351 management to a capable Master by issuing a Broadcast Command Code **Disable Slave Events Command**
1352 **(DISEC)** to disable Hot-Join by setting the DISHJ bit (see *Section 5.1.9.3.1*).

5.1.8 Timing Control

1353 I3C includes optional Timing Control and Timestamping of events generated by I3C Devices resident on the
1354 I3C Bus. This Timing Control framework allows uncertainties affecting the transmission or reception of
1355 timing information (for example: Bus activity, busy Master or Slave, operation latency, system jitter, etc.) to
1356 be nullified.

1357 The I3C Timing Control framework provides:

- 1358 • Flexible implementation with significant capability enhancements
- 1359 • Means for synchronizing the timing references/clocks, and subsequently the events, of Slave
1360 Devices on the I3C Bus to the timing reference/clock of the Master

1361 This can result in reduced energy consumption at the system level.

- 1362 • Transmission of timing information, with minimal complexity for the Slave Devices
- 1363 • Controllable timing accuracy, suitable for the target use cases

1364 I3C defines two forms of systems and events for Timing Control. Only one form can be used on the I3C Bus
1365 at any time:

- 1366 • **Synchronous Time Control** (see *Section 5.1.8.2*): The Master emits a periodic time sync, which
1367 allows all Slaves to set their sampling time relative to this time sync. This avoids drift of individual
1368 Slaves' clocks. It also ensures that samples occur close together in time, permitting data collected
1369 at the same time to be fused. It also provides a calibration method, enabling more accurate
1370 sampling between the time syncs.
- 1371 • **Asynchronous Time Control** (see *Section 5.1.8.3*): Slaves timestamp the moment at which they
1372 acquire sampled data, permitting the Master to time-correlate samples received from multiple,
1373 different sensors in an easier and more accurate manner. Timestamping ensures that the Master
1374 always has the time at which sensor data acquisition occurred (subject to the timing granularity
1375 supported), even if there are latencies in moving the data from the Slave to the Master. Four
1376 different Asynchronous Time Control versions ('Modes') provide different methods for measuring
1377 time, and/or calibrating timing, with increasing accuracy.

5.1.8.1 General Principles

1378 The exchange of timing information is based on agreements between the Master and Slaves.

1379 The elements of this agreement are:

- 1380 1. Two I3C Common Command Codes:
 - 1381 • Set Exchange Timing Information (SETXTIME), see *Section 5.1.9.3.20*, and
 - 1382 • Get Exchange Timing Information (GETXTIME), see *Section 5.1.9.3.21*.

1383 These CCCs:

- 1384 • Identify Bus events and markers (i.e., specific SDA edge and/or SCL edge),
 - 1385 • Transfer timing or control information (e.g., 'abort' command),
 - 1386 • Specify which timing control procedure is in effect, and
 - 1387 • Query the Slave Device for Exchange Timing support details, such as which Timing Control
1388 Mode(s) are supported, the current Timing Control state, frequency, inaccuracy, etc.
- 1389 2. One defined and Mode-specific marker is sent by the Master, and used to synchronize the Slaves
 - 1390 3. One data collection event is sent by the Slave. It includes the time information previously
1391 requested by the Master.

5.1.8.2 Synchronous Systems and Events

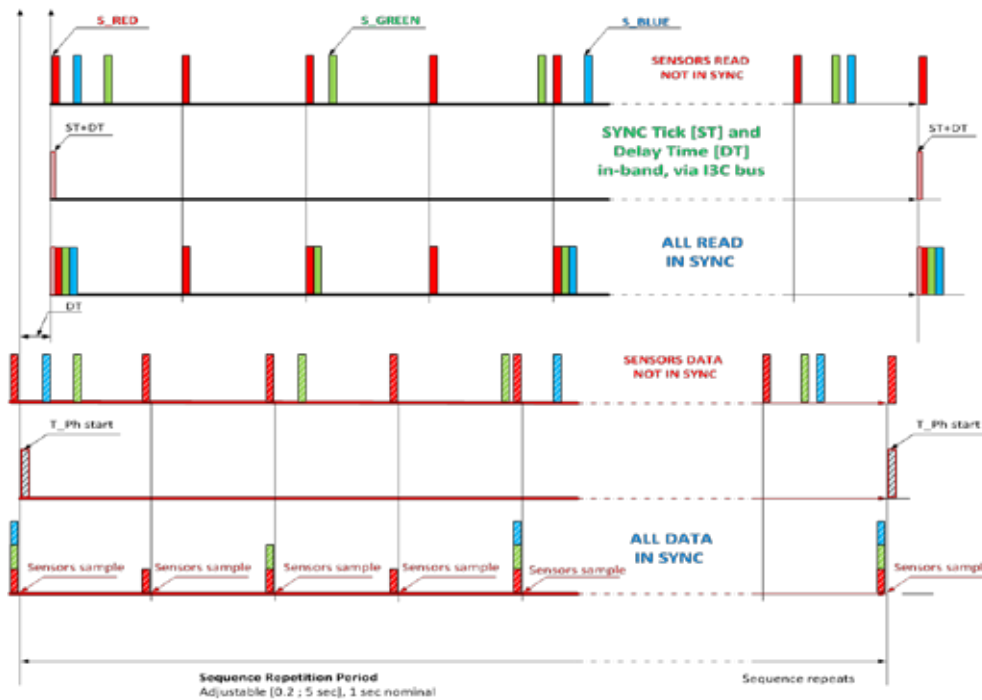
1392 In systems where multiple sensors (or other Slave Devices) provide periodically sampled data, it is
 1393 advantageous to instruct the Slaves to be able to collect the data at essentially synchronized times, so that the
 1394 Master can read several Slaves' data in a single system awake period.

1395 In a typical system, different Slaves will sample their data at different, uncorrelated times. This is true even
 1396 if all Slaves are set to the same sampling frequency, because Slave-to-Slave oscillator accuracy differences
 1397 will cause drift over time. In I3C's Synchronous Time Control mechanism, the Master periodically emits a
 1398 synchronization pulse, called a SYNC Tick. This method permits all Slave data sampling to occur very close
 1399 together in time, so that Slaves can prepare and activate their individual sampling mechanisms, even if there
 1400 are variances across their clocks.

1401 Terminology

- 1402 • **S_RED, S_GREEN, S_BLUE:** Three different Slaves. In the Figures below, color-coded
 1403 rectangles depict either data availability (striped rectangle) or a read of their data (solid rectangle)
 1404 for the corresponding Slave/Sensor.
- 1405 • **Examples:** A striped blue rectangle indicates S_BLUE has data available. A solid green rectangle
 1406 indicates an S_GREEN data read.
- 1407 • **T_Ph:** A pre-established, Master-provided time duration that the Slaves use to adjust both (a) their
 1408 internal timers, and (b) the time at which a sequence of sampling events begins.
 - 1409 • **T** is for 'time period'. The sequence of sampling events must be completed within this period
 1410 of time, across all of the Slaves.
 - 1411 • **Ph** is for 'phase'. The sequence of sampling events must begin at the same moment, across all
 1412 of the Slaves.
- 1413 • **SYNC Tick [ST]:** An I3C SETXTIME CCC with ST Sub-Command sent by the Master, selecting
 1414 a specific hardware event on the I3C Bus (from among a multitude of other, similar events) as the
 1415 timing reference. Slaves shall use the time moment at which the SYNC Tick event takes place to
 1416 determine the start and end times of T_Ph.
- 1417 • **Delay Time [DT]:** An I3C SETXTIME CCC with DT Sub-Command sent by the Master,
 1418 providing the elapsed time between the correct beginning of the current T_Ph and the ST-
 1419 designated hardware event, using the exact moment at which the hardware event actually takes
 1420 place on the I3C interface. The Slaves shall use DT to determine the precise moments for the T_Ph
 1421 start and end times, and to correct their timers' accuracy.
- 1422 • **ODR:** Output Data Rate, a parameter specific to each Slave that indicates its constant data rate
 1423 (i.e., the number of samples taken by its sensors per a given period of time).

1424 **Figure 21** depicts a system with initially unsynchronized sampling timing, and its transition to a synchronized
1425 system.



1426
1427

Figure 21 Synchronization of Sampling Moments

1428 The top three baselines (in black) of **Figure 21** show I3C Bus activity, and the bottom three baselines (in red)
1429 show internal data availability for each individual Slave with connected sensors:

- 1430 1. The top black and red baselines show an unsynchronized System. The Slaves are all running at
1431 different ODRs, and are all using totally unrelated timers.
- 1432 2. The middle black and red baselines show the Master sending time synchronization information,
1433 and the Slaves subsequently processing that time sync information.
- 1434 3. The bottom black and red baselines show a synchronized System. The Slaves collect their sensor
1435 data at synchronized times, and the Master reads all Slaves' data during a single system awake
1436 period.

1437 The synchronized System adjusts both the frequency and the phase of the Slaves' sampling periods. The
1438 Master sends the synchronization information periodically, with a repetition period of T_{Ph} (a rather large
1439 time interval).

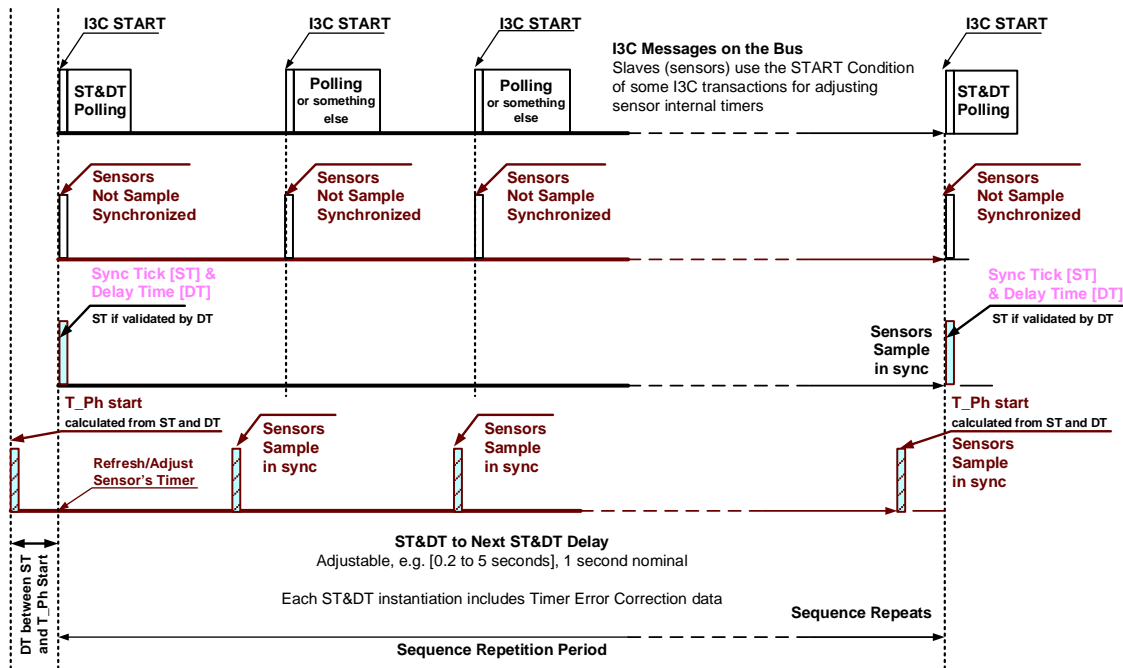
1440 Ideally, T_{Ph} is exactly divisible by the Least Common Multiple of the sampling periods of all Slaves active
1441 on the I3C Bus. In typical cases, however, the LCM is often not a practical choice due to variations in ODR
1442 and sampling period among the several Slaves. In such situations, the application could either adjust some of
1443 the ODRs, or as a last resort, synchronize Slaves in smaller groups of better mutual compatibility.

1444 T_{Ph} Start (i.e., the moment that phase-aligns the beginning of the sampling sequence across all the Slaves)
1445 shall correspond to the time at which most of the Slaves would collect sensor data simultaneously.

1446 During a period of T_{Ph} , all of the Slaves will have updated data available at least once.

1447 T_{Ph} should be such chosen that the Slaves' timers can maintain the required accuracy. For many general
1448 applications, a T_{Ph} period of 1.0 second would be appropriate.

1449 **Figure 22** depicts a Synchronization Procedure on a general-purpose I3C Bus, as detailed below.



1450

1451

Figure 22 Synchronization Procedure on a General-Purpose I3C Bus

1452 This Synchronization Procedure relates to the Timing Control General Principles as follows:

- 1453 • The mutually identifiable hardware event is the Start Condition's SDA falling edge.
- 1454 • The Command Codes are two SETXTIME CCCs (see [Section 5.1.9.3.20](#)): One with the Sync Tick
- 1455 (ST) Sub-Command, the other with the Delay Time (DT) Sub-Command.

1456 The Synchronization Event takes place periodically, at intervals of T_{Ph} . This repetition time is
 1457 adjustable, and is determined by the Master. In this example, a period of 1.0 second has been
 1458 chosen (i.e. the Synchronization Event repeats at 1.0 second intervals).

1459 The Synchronization Procedure consists of the following steps:

- 1460 1. Each Synchronization Event consists of one ST Message, followed by one DT Message:
 - 1461 • The ST Message shall be sent as soon after a START Condition (and, for a Direct Message,
 - 1462 after the Slave Address) as possible, in order to provide enough time for the DT Message to be
 - 1463 sent and received.
 - 1464 • The DT Message shall arrive before the next of the shortest data polling time window.
 - 1465 • The DT Message shall contain either (a) a valid Time Delay between the START Condition and
 - 1466 the required T_{Ph} Start, or else (b) an 'abort' order for the current sync window (T_{Ph}), as
 - 1467 determined by the value of the Defining Byte's most significant bit (see [Table 55](#)).
- 1468 2. Each Slave shall record the value of its internal timer/counter at the moment when the START
- 1469 Condition's SDA falling edge is detected on the I3C Bus. A record of the time of last START
- 1470 Condition (and only the last START Condition) shall be stored in a shadow register.
- 1471 3. Upon recognizing both (a) either its own Slave Address, or the Broadcast Address (7'h7E), and (b)
- 1472 the ST Message (i.e., SETXTIME CCC with ST Sub-Command), each Slave shall use the stored
- 1473 START Condition time as a reference for the beginning time of the new T_{Ph} period.
- 1474 4. Upon recognizing the subsequent DT Message (i.e., SETXTIME CCC with DT Sub-Command),
- 1475 each Slave shall either:

- 1476 a. Correct the T_Ph beginning time and T_Ph duration (if needed) with respect to the internal
1477 timer,
1478 Or else:
1479 b. Abort the current Synchronization Procedure, preserving the internal timer's running
1480 parameters.
1481 5. When the T_Ph interval expires (i.e., after approximately 1.0 second in our example), the Master
1482 shall repeat the Synchronization Event. That is, the Master shall again send first the ST Message
1483 and then the DT Message in the manner described in steps 1 through 4 above.

1484 The above Synchronization Procedure requires three Messages to configure the Slaves:

1485 **1. ODR (Output Data Rate) Message**

1486 This Message sets the Slave's running Output Data Rate. The Message is specific to each Slave,
1487 and is not necessarily an I3C CCC.

1488 If the I3C SETXTIME CCC (see *Section 5.1.9.3.20*) is used as the ODR Message, then its
1489 Defining Byte field will contain the ODR Sub-Command (0x8F), followed by one byte of Sensor-
1490 specific ODR selection Data.

1491 **2. TPH (Procedure Repetition Time) Message**

1492 This Message sets the duration of T_Ph, the Synchronization Event repetition period. The Message
1493 is specific to each Slave (or to all Slaves), and is not necessarily an I3C CCC.

1494 If the I3C SETXTIME CCC (see *Section 5.1.9.3.20*) is used as the TPH Message, then its
1495 Defining Byte field will contain the TPH Sub-Command (0x3F), followed by one or two data
1496 bytes, as agreed between the Master and the Slave.

1497 **3. TU (Time Unit) Message**

1498 This Message sets the value of the time unit transferred to the Slave. The Message is specific to
1499 each Slave (or to all Slaves), and is not necessarily an I3C CCC.

1500 If the I3C SETXTIME CCC (see *Section 5.1.9.3.20*) is used as the TU Message, then its Defining
1501 Byte field will contain the TU Sub-Command (0x9F), followed by one byte of Slave-specific TU
1502 Data.

5.1.8.3 Asynchronous Systems and Events

1503 This Section defines four variations, called Asynchronous Timing Control Modes, on a common procedure
1504 for increasing the precision of an acquired time-of-occurrence of an event in a sensor or other I3C Slave
1505 connected to an I3C Master. The four Asynchronous Modes are presented in order of increasing accuracy
1506 and/or precision.

1507 The actual event itself may or may not be periodic, and the Master and Slave devices need not use the same
1508 time base clock (source, frequency or accuracy). The procedure can help address the large
1509 inaccuracies/offsets to the event's acquired time-of-occurrence that IBI launch/acknowledge latencies can
1510 introduce.

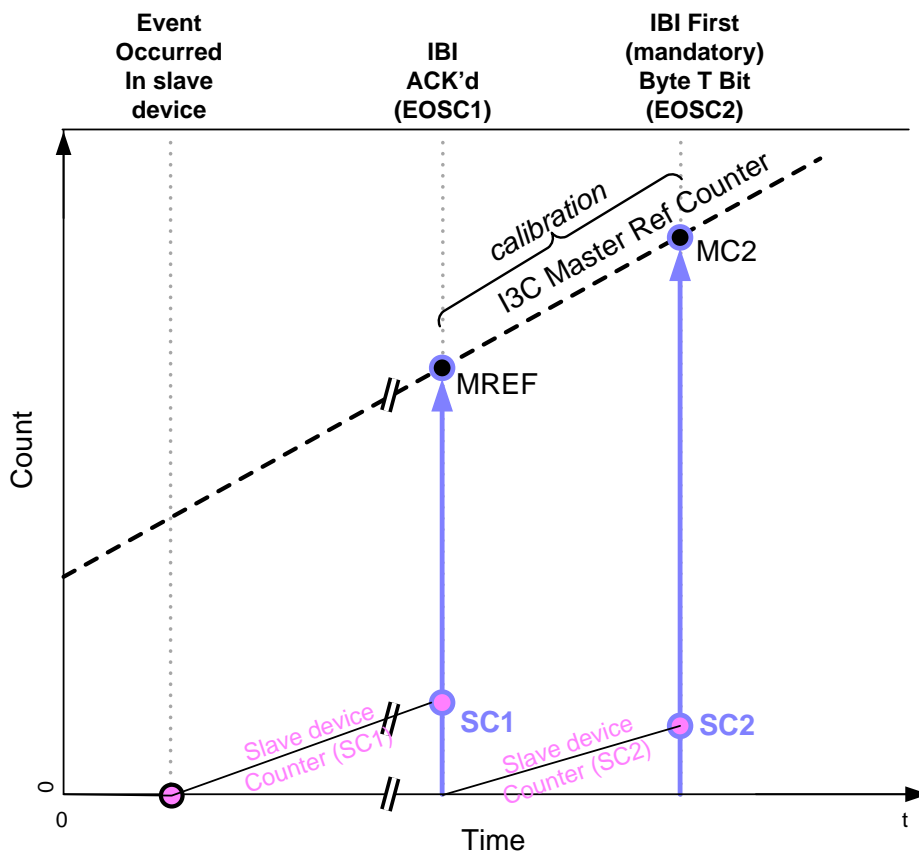
1511 In I3C Asynchronous Timing Control, a Slave Device timestamps events occurring within that Slave, and
1512 then notifies the Master about it by generating an In-Band Interrupt (IBI). The Slave's IBI may not reach the
1513 Master immediately – either due to the I3C Bus being busy, or because the Slave lost IBI arbitration, or
1514 because the Master did not immediately accept the IBI. The Asynchronous mechanism allows the I3C Master
1515 to convert the Slave's timestamp value to its own internal time scale, in terms of both timing resolution and
1516 timing accuracy.

1517 Terminology

- 1518 · **SC1:** Slave counter from Event occurrence to a Master-initiated end-of-count (EOSC1) point
1519 depending on Mode (e.g. IBI Acknowledged point in Mode 0 & Mode 1).

- 1520 · **SC2:** Slave counter from EOSC1 to a second Master-initiated end-of-count, EOSC2 (e.g. T-Bit of
- 1521 IBI Mandatory byte in Modes 0 & 1).
- 1522 · **aME:** Optional additional Master-initiated bus Events, recognizable by the Slave, used to both
- 1523 form the end-of-SC1 (EOSC1), and to provide a countable reference ‘tick’ (which is reported as
- 1524 aME_TICKS, used in Mode 1). This event could be either periodic or aperiodic. It is used to help
- 1525 reduce the run-length of the counters.

1526 **Figure 23** illustrates the I3C Asynchronous Timing Control model. In this model, the I3C Master Device
 1527 maintains its own notion of time (‘I3C Master Ref Counter’ in the Figure) using whatever timing source and
 1528 units it needs; for example, a real-time clock, or a simple 32-bit rollover timer/counter. This notion of time
 1529 allows the Master to determine the relative timing among multiple events generated by different Slaves.
 1530 These relative time differences can then be used to correlate the Slave-generated events, for example to
 1531 support sensor fusion, to plot sensor events vs. absolute time, or for other purposes.



1532

1533 **Figure 23 Graphical Representation of Async Mode 0 Timestamp Interpolation**

1534 I3C Slave Devices report the occurrence of sensor events to the I3C Master using IBI, but because the IBI
 1535 can be delayed before reaching the Master, the Master’s notion of time alone is not sufficient to accurately
 1536 gauge event timing. For this reason, the Slave maintains its own counter (‘Slave Device Counter (SC1)’ in
 1537 **Figure 23**), driven by its own clock. At the time when the Slave is able to generate the IBI to report the
 1538 occurrence of a sensor event to the I3C Master, the number of Slave clocks elapsed since the event occurred
 1539 (‘SC1’ in the Figure) and the calibration period (SC2) shall be sent as IBI payload of the same IBI.

1540 The Slave Device counts can be converted into Master time via a Master-provided method whereby the Slave
 1541 can measure the time between two points (‘EOSC1’ and ‘EOSC2’ in the Figure) as SC1, in terms of number

1542 of its clock cycles whose approximate clock frequency is known to the Master through the GETXTIME CCC,
1543 see **Section 5.1.9.3.21**). The Slave then counts the time (SC2) from EOSC1 to EOSC2, and then
1544 communicates both SC1 and SC2 to the Master as IBI payload. This allows the Master to correlate SC1 and
1545 SC2 with its own internal counter values MREF and MC2, respectively. The Master knows the center clock
1546 frequency of the Slave Device (again via the GETXTIME CCC), and can compare the Slave-provided count
1547 with its expected value, given the known length of time reported by SC2. The delta between the expected
1548 count and the actual count can be used as guidance to improve the accuracy of the SC2 value, derived from
1549 the GETXTIME CCC, and thereby the accuracy of the event's calculated time-of-occurrence.

1550 Note that the higher the frequency of the Slave clock vs. the effective SCL clock frequency during the SC1-
1551 to-SC2 measurement, the more accurate the error delta will be. This means that the Master could gain more
1552 accuracy in the scale factor by extending the SC1-to-SC2 period.

1553 The Slave shall report the SC1 value as a 2-byte (16-bit) value, transmitting the Least Significant Byte first,
1554 and then the Most Significant Byte. The Slave shall report the SC2 value as a 1-byte value. If either count
1555 overflows (which can happen with a busy Bus with a large transaction), then the Slave shall report values of
1556 all 1'b1 bits: 8'hFF for SC2, and 16'hFFFF for SC1.

1557 A Slave may choose to not count SC2 value if its counter's clock source is insufficiently accurate, however
1558 in this case it shall report a value of 8'h00 for SC2 in the IBI payload. A Master receiving an SC2 value of
1559 8'h00 shall rely solely on the frequency value reported by GETXTIME for scaling. Slaves having very stable
1560 clocks, especially when fast enough, could help the Master achieve higher time-stamping accuracy by
1561 supporting and reporting the SC2 count.

1562 In all four Asynchronous Modes, the key points are:

- 1563 1. All I3C Devices supporting any of the Asynchronous Modes shall drive a Mandatory Byte after
1564 Acknowledgement of their IBI. As a result, bit [2] of the BCR for these I3C Devices shall be set
1565 (i.e., shall have the value 1'b1).

1566 In the Mandatory Byte:

- 1567 · Bit [7] shall be set to 1'b1, indicating that a timestamp follows
- 1568 · Bits [6:0] are set by the Slave and read by the Master, and are interpreted according to a private
1569 contract between them. These bits are usually used to convey some aspect of the event.

- 1570 2. Two mutually identifiable Events on the I3C Bus between I3C Master and I3C Slave Devices:

- 1571 · **End of SC1 (EOSC1):** The SC1 latch point and the start of the Calibration Window to count
1572 SC2.

1573 The actual event used depends on which of the four Asynchronous Modes is being used.
1574 It could be either the IBI ACK point of the current IBI transaction, or an SCL edge, or
1575 some other event on the I3C Bus.

- 1576 · **End of SC2 (EOSC2):** The end of the Calibration Window.

1577 This event depends on which of the four Asynchronous Modes is being used. It could be
1578 either the T-bit following the IBI's Mandatory Byte for the current IBI transaction, or an
1579 SCL edge, or some other event on the I3C Bus.

- 1580 3. If desired, higher accuracy in the time-stamp counters can be achieved by using additional,
1581 optional common Events (**aME**) on the I3C Bus, when the clock frequencies available to the
1582 Device are less accurate.

- 1583 · These optional aME events could be either SCL edges, START Condition events, or Repeated
1584 Start events. I3C Slave Devices supporting timestamped events in this mode shall (in addition
1585 to running their internal SC1 & SC2 timing counters after sampling each sensor event)
1586 maintain a count of the aMEs that they receive on the I3C Bus, and shall send all these counter
1587 values as the payload for their IBIs. The count of aMEs from the event to the IBI accept point
1588 is one byte wide, and is sent to the Master as the fourth byte of the IBI payload (i.e., after the
1589 SC2 value).

- 1590 4. After the Mandatory Byte, the Slave sends its timestamp values:
1591 • **SC1:** The 16-bit timestamp of the original sensor-generated event
1592 • **SC2:** An 8-bit reference/calibration count
1593 • **aME_TICKs:** A 1-byte value indicating the number of aMEs that the Slave has detected, from
1594 the sensor event to the IBI-ACK.

1595 The values of both SC1 and SC2 are in terms of the Slave's time scale. The Master will convert
1596 these values into its own time scale, taking into consideration the Slave's counter clock frequency
1597 as reported by the GETXTIME CCC.

1598 In order to support a wide range of applications, several degrees of accuracy are required for such
1599 timing related information. As a result, I3C provides four different Asynchronous Modes which
1600 are summarized in *Table 13* and detailed in the following subsections. Each of these methods
1601 addresses a different level of accuracy, and presents a distinct complexity tradeoff option for the
1602 Master and/or Slave. The SETXTIME CCC (see *Section 5.1.9.3.20*), is used to enter all four
1603 Async Modes, using the 'Defining Byte' (Sub-Command) field to select the desired Mode.

1604

Table 13 Asynchronous Timing Control Modes

	Async Mode 0	Async Mode 1	Async Mode 2	Async Mode 3
	Asynchronous Basic Mode	Asynchronous Advanced Mode	Asynchronous High-Precision Low-Power Mode	Asynchronous High-Precision Triggerable Low-Power Mode
See Section	5.1.8.3.1	5.1.8.3.2	5.1.8.3.3	5.1.8.3.4
Description	Simple for I3C Master and I3C Slave Allows all I2C out-of-band interrupt usages to be replaced with I3C In-Band Interrupt	Extension to Async Mode 0 If clock used by Slave's counter drifts, or is not accurate enough, then the Master must send an aperiodic event to limit the running time of the Slave's timing counter.	Minimizes the time for which the clock driving the Slave's timer counter needs to run. Devices can use burst clock sources, at the cost of Master overhead to measure time and correlate time scales. Time reference is falling SCL when Master ACK's IBI	Precision controlled trigger followed by precision time-stamp of detected event. Time stamp similar in operation to Mode 2, except time reference is to sync signal preceding trigger.
Required	Mandatory for Master Optional for Slave	Optional		
SETXTIME Sub-Command to Enter Mode Defining Byte Value	Enter Async Mode 0 8'hDF	Enter Async Mode 1 8'hEF	Enter Async Mode 2 8'hF7	Enter Async Mode 3 8'hFB
Clock Used for Slave's Timer Counter	Local clock in Slave		SDR, DDR Mode: SCL falling edges TSL, TSP Mode: SDA SCL transitions Typically includes additional burst counter ('B-count'), which can be driven by a burst oscillator.	Trigger: SDR, DDR Mode (SCL falling edges) Time Stamp: Clock like Mode 2
Sensor Sample	T0			
EOSC1	In-Band Interrupt ACK Bit		First falling edge of SCL after the Sensor Sample event.	
EOSC2	T-bit following the Mandatory Byte after ACK of In-Band Interrupt		Second SCL falling edge after Sensor Sample event.	
aME	N/A	I3C SDR START	SCL edge	

1605

5.1.8.3.1 Async Mode 0: Asynchronous Basic Mode

1606 All I3C Master Devices capable of Asynchronous Timing control procedures shall support Async Mode 0,
 1607 Basic Asynchronous Timestamping. This Mode expects that a reasonably accurate and stable clock source is
 1608 available in the Slave to drive the timer counter. Note that in this Mode, the two SCL edges both occur after
 1609 the Slave and the Master agree that the transaction was valid.
 1610 To select Async Mode 0, the Master sends the SETXTIME CCC (see *Section 5.1.9.3.20*) with the Sub-
 1611 Command Enter Async Mode 0 (0xDF).

Terminology

- 1613 · **EOSC1**: First SCL positive-going edge after the IBI ACK for the Slave Device
- 1614 · **EOSC2**: First SCL positive-going edge after the T-Bit of the IBI's Mandatory Byte
- 1615 · **aME**: Not used

1616 *Figure 23* graphically depicts Async Mode 0 timestamp data transfer from the Slave to the Master, allowing
 1617 the Master to calculate the time at which sensor data sampling has taken place. It is presumed that the Slave
 1618 is able to transfer the data within a time interval that the Slave's timing counter can meaningfully measure
 1619 (i.e., that the clock driving the timing counter preserves a sufficiently stable frequency to increment it in a
 1620 substantially linear manner with respect to time).

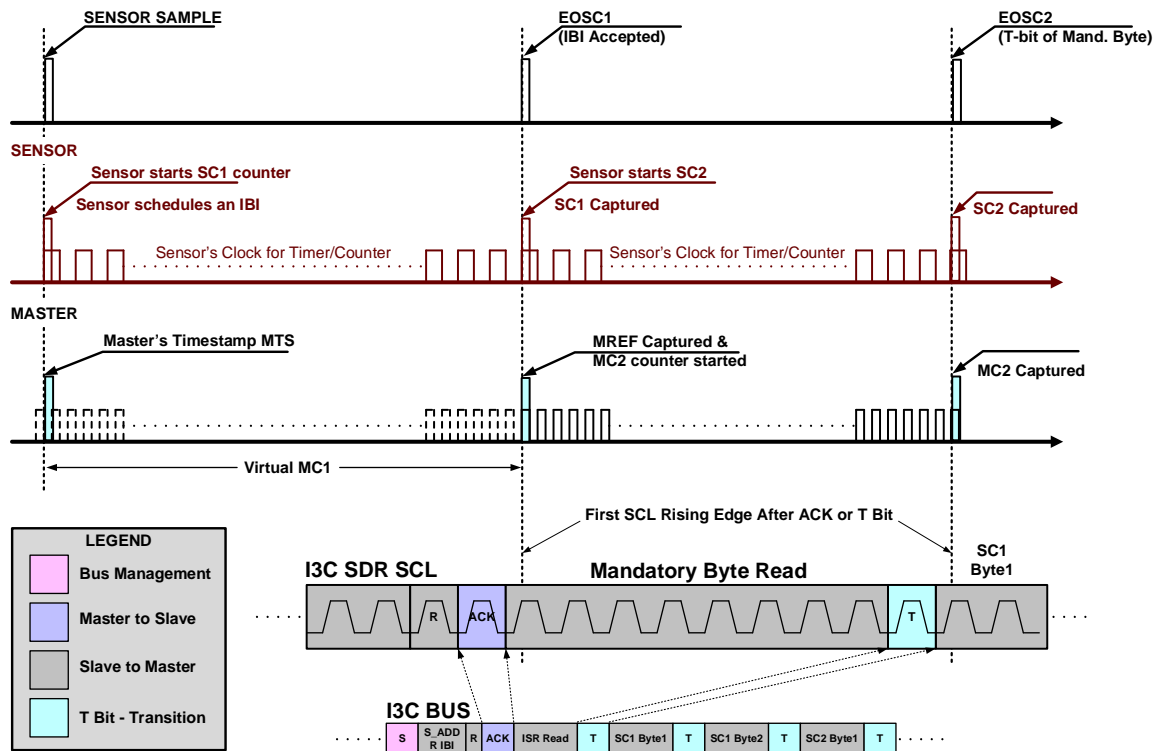


Figure 24 Example of Asynchronous Mode 0 Timestamp Data Transfer

1623 *Figure 24* illustrates the basic sequence of operations during a simple asynchronous time information
 1624 transfer:

- 1625 1. The top black baseline shows the events visible outside of the Slave and the Master
- 1626 2. The middle (red) baseline shows the activity inside the Slave's timing counters
- 1627 3. The bottom black baseline presents the virtual and real activity inside the Master's timing counter

1628 In more detail, **Figure 24** illustrates that:

- 1629 1. At sensor sampling/event time, the Slave shall start its internal timing counter, and initiate the IBI
1630 request.
- 1631 2. If the IBI request is accepted, then:
- 1632 a. Immediately:
- 1633 · The Slave shall record its internal timing counter, SC1
 - 1634 · The Master shall record its internal timing counter, MREF
- 1635 b. At the T-Bit of the IBI payload's mandatory byte:
- 1636 · The Slave shall record its internal timing counter, SC2
 - 1637 · The Master shall record its internal timing counter, MC2
- 1638 c. The Slave shall transfer the recorded SC1 & SC2 values in the prescribed order.
- 1639 3. If the IBI request is not accepted, then:
- 1640 · The Slave shall continue to increment its own counter for SC1
 - 1641 · The Slave shall wait for the next opportunity for the IBI request to be issued & accepted
 - 1642 · When the IBI request is eventually accepted, the Slave shall proceed as described in step 2
1643 above.
- 1644 4. Evaluating the timing counter data:
- 1645 If the numbers are non-zero, then the Master calculates the real time interval based on its internal
1646 timing counter, using the formula:

1647
$$MTS = MREF - MC2 \times SC1/SC2$$

- 1648 · If SC1 is zero, then no time information is assigned to the event
- 1649 · If SC2 is zero, then the scaling shall be done based on the slave device counter's clock
1650 frequency as can be read with GETXTIME CCC.

1651 Note that the above method of recovering the target event's timestamp is only suitable for periods during
1652 which the Slave clock is appropriately stable. The calibration time available is the duration elapsed between
1653 EOSC1 and EOSC2, which in this mode is 9-bit durations on the I3C Bus for the Mandatory Byte. To enable
1654 better calibration for Slaves using low-frequency clocks, the Master may optionally use clock stalling during
1655 the Mandatory Byte transfer, thus lengthening the calibration window; however, clock stalling has the
1656 disadvantage that IBIs will consume more I3C Bus time.

1657 If better calibration is required and clock stalling is not an option, then other, more refined procedures or
1658 other Async Modes can be used.

5.1.8.3.2 Async Mode 1: Asynchronous Advanced Mode

1659 Async Mode 1 is an extension of Async Mode 0 that retains EOSC1 and EOSC2, but achieves much higher
 1660 timing accuracy by using a mutually identifiable additional Bus Event (aME), which could be periodic or
 1661 aperiodic. This aME helps in restricting the run of the internal timer counters inside the Slaves. For Async
 1662 Mode 1 the aME event is the I3C SDR Start event. SDR Start is used because it is visible on the I3C Bus to
 1663 both the Master and the Slave. Repeated START is not an aME in Async Mode 1.

1664 After the sensor event occurs, the Slave shall increment its SC1 counter until it detects the first aME (I3C
 1665 SDR Start). The Slave shall count the number of aME events it detects until it receives an ACK to its IBI
 1666 request, and then shall send (after the IBI Mandatory Byte) the value of its internal timer counters SC1 and
 1667 SC2 (just as in Mode 0), followed by the value of the aME event counter (aME_TICKS). Using this
 1668 information, the Master can accurately derive the time of the sensor event.

1669 To select Async Mode 1, the Master sends the SETXTIME CCC (see *Section 5.1.9.3.20*) with the Sub-
 1670 Command Enter Async Mode 1 (0xEF).

1671 *Figure 25* illustrates the operation of Async Mode 1 for timestamping, including the data transfer sequence.

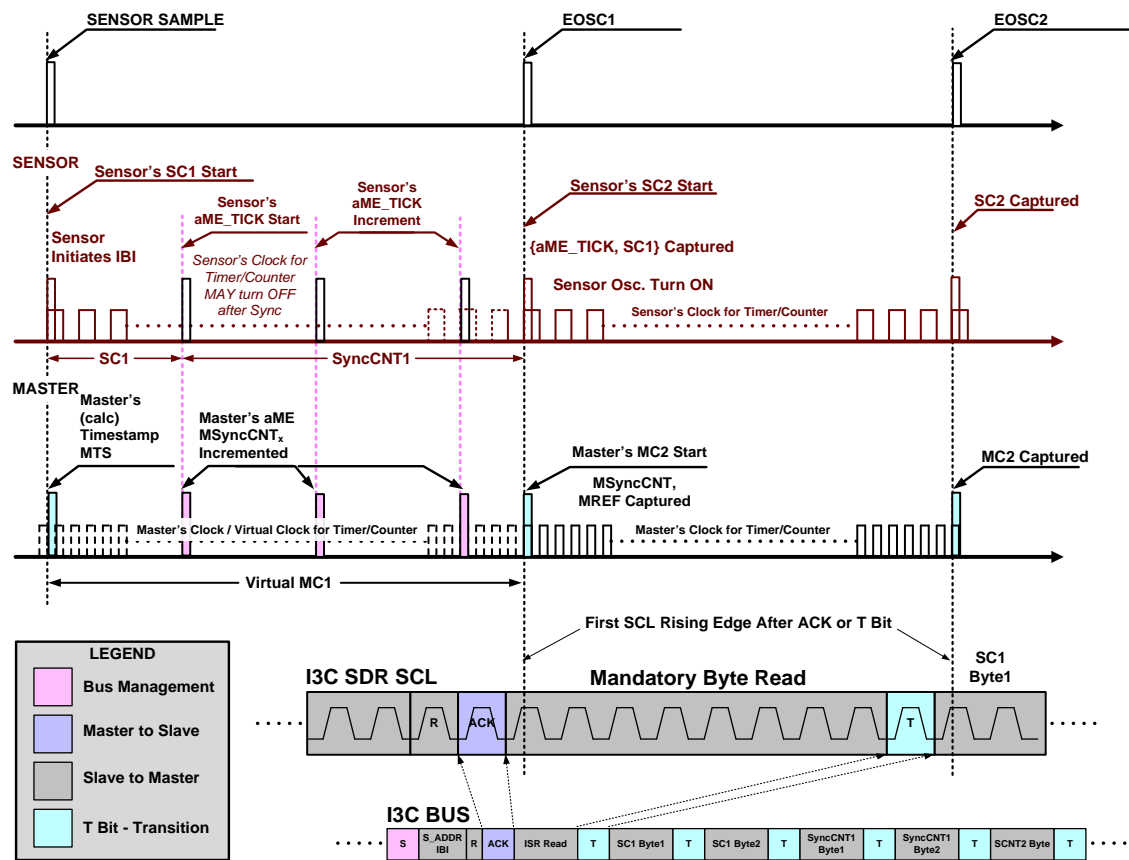


Figure 25 Asynchronous Mode 1 Timestamp Data Transfer

1672

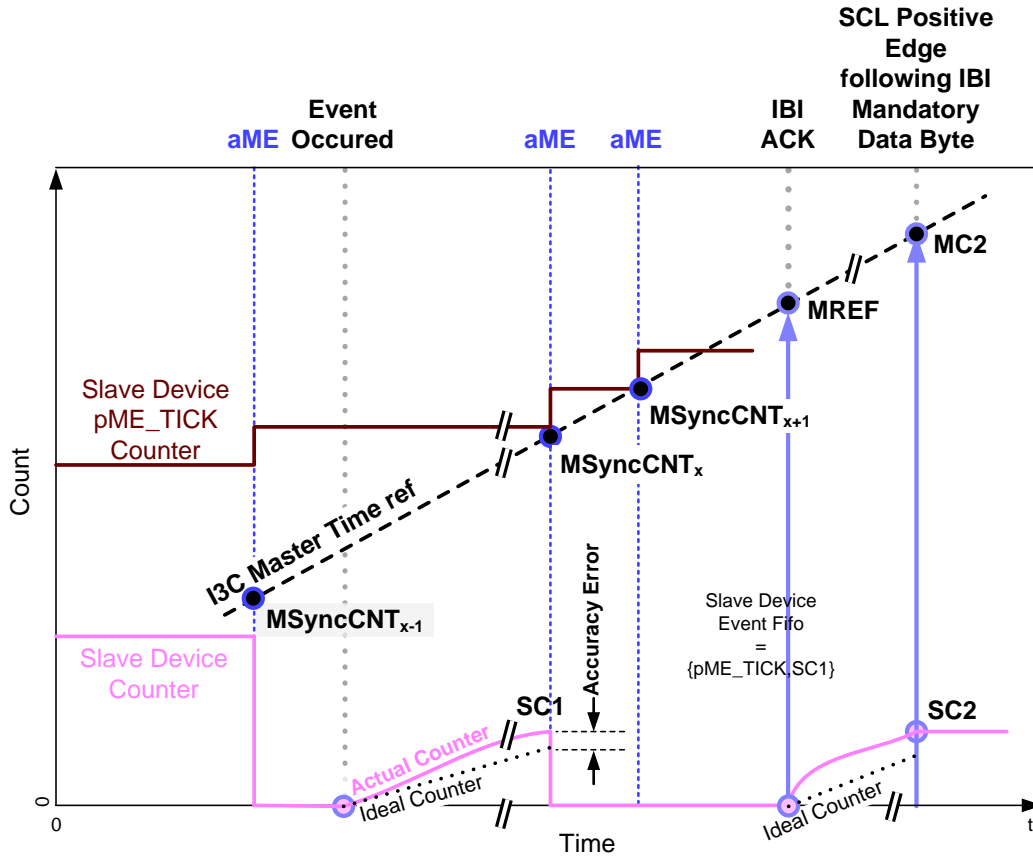
1673

1674 *Figure 26* shows an example of the Master's interpolation of Slave-generated timestamps using the Master's
 1675 internal time scale.

1676 The horizontal axis represents elapsed time, and the vertical axis shows the values of three separate counters:

- 1677 · The Master's internal timer counter (black dashed line) which increments steadily,
- 1678 · The Slave's internal timer counter (pink solid line), and
- 1679 · The Slave's internal aME event counter (brown solid line) which steps at each aME event.

1680 Events that are visible on the I3C Bus, including EOSC1, EOSC2, and aME events, are indicated by blue
 1681 vertical dashed lines. Note that the Slave's counter starts incrementing from zero when a sensor event occurs
 1682 ('Event Occurred'), captures SC1 on first aME ('SCL Posedge'), and resets to zero at each aME.



1683

1684

Figure 26 Async Mode 1 Timestamp Interpolation

5.1.8.3.3 Async Mode 2: Async High-Precision Low-Power Mode

1685 Async Mode 2 supports high resolution time measurements, and is ideal for systems with multiple Slaves
 1686 where higher precision and lower power consumption is needed. Higher precision is achieved by using SCL
 1687 falling edges as a common coarse timing reference for Master and Slave. Low power is achieved because the
 1688 Slave need not generate its own clock. Extending resolution requires each Slave to make use of a burst
 1689 oscillator to interpolate the time between a detected event and the next SCL falling edge. Because the burst
 1690 oscillator is only turned on for a short period after a detected event, total energy consumption is lower than
 1691 for any of the other Async Modes.

1692 To select Async Mode 2, the Master sends the SETXTIME CCC (see *Section 5.1.9.3.20*) with the Sub-
 1693 Command Enter Async Mode 2 (0xF7).

1694 The hardware implementation of Async Mode 2 in the Slave is simple; this is achieved at the expense of
 1695 somewhat greater complexity on the Master side. Since there are usually many more Slaves than Masters,
 1696 this is considered a reasonable trade-off. Also, Slave logic (as contrasted with Master logic) often uses a
 1697 coarser technology node which, although well-suited for the higher-voltage operation that sensors often
 1698 require, has a higher cost per gate, both in terms of energy and in terms of silicon area.

1699 In Async Mode 2 the Slave timestamp reference clock (SCLK) is derived from the I3C Bus itself (see *Figure*
 1700 *28*), in a manner that depends on the Bus Mode currently in use:

- 1701 · **I²C Mode, I3C SDR Mode, and I3C HDR-DDR Mode:** SCL is used as SCLK
- 1702 · **I3C HDR-TSL Mode and HDR-TSP Mode:** All transitions (SDA, SCL and both simultaneously)
 1703 generate SCLK

1704 *Figure 27* and *Figure 29* illustrate that in operation, when the Slave detects a sensor event, the Slave shall
 1705 turn its burst oscillator on, and shall enable two counters (which shall have been cleared to zero previously):

- 1706 · **B-Count:** Increments on burst oscillator cycles
- 1707 · **S-Count:** Increments on SCLK falling edges.

1708 The Slave shall also initiate an IBI at the earliest opportunity (i.e., first Bus Available condition). On the first
 1709 SCLK falling edge after the detected event, the value of B-Count shall be loaded into register C1. On the
 1710 second SCLK falling edge, the value of B-Count shall be loaded into register C2. At this time, the burst
 1711 oscillator shall be turned off, and B-Count may optionally be cleared. Ultimately, when the IBI is
 1712 acknowledged, the value of S-Count shall be loaded into register C0, while the Slave shall transfer the
 1713 contents of registers C1, C2, and C0 to the Master.

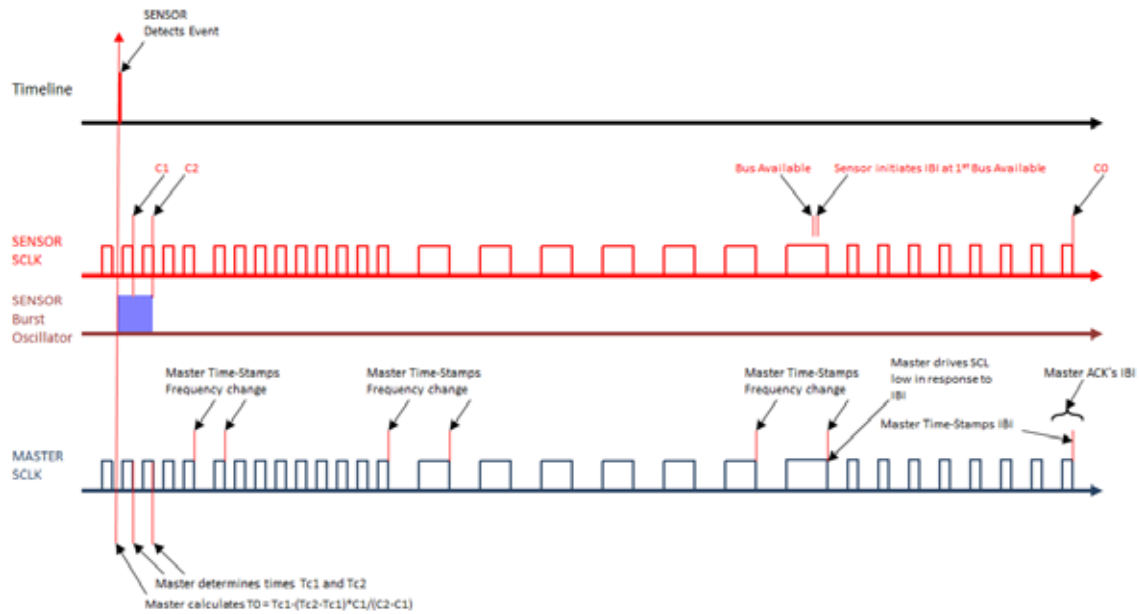
1714 The Master shall timestamp (i.e., against its own time-base) the time at which it ACKs the IBI from this Slave
 1715 (specifically, the time of the SCL falling edge during the ACK). The Master will have similar hardware as
 1716 the Slave for generating SCLK, and shall maintain an S-Count counter which shall run continuously while
 1717 in Async Mode 2. The Master shall also timestamp (and data-log) its S-Count whenever a change in SCLK
 1718 frequency occurs. From the count C0 and its own data-log, the Master will be able to reconstruct the time at
 1719 which C1 and C2 occurred (Tc1 and Tc2, respectively). The Master shall then calculate the time at which the
 1720 Slave detected the event, using the following formula:

$$1721 \quad T_0 = T_{c1} - (T_{c2} - T_{c1}) * C1 / (C2 - C1)$$

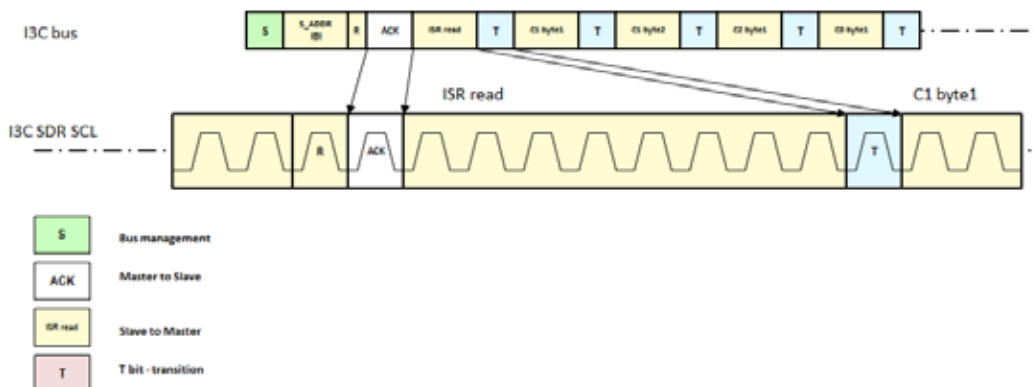
1722 Note that because the Master generates SCL (for I²C mode, and for I3C SDR and HDR-DDR Modes), SCL
 1723 is likely to be a rational fraction of the Master's internal timebase. This should result in a very compact and
 1724 accurate timestamp relative to this timebase. For I3C HDR-TSL and HDR-TSP Modes accuracy degrades
 1725 somewhat because the Master doesn't generate the timebase, however high precision among multiple sensors
 1726 will still be maintained because the timebase is shared (as contrasted with being independently generated in
 1727 each Slave).

1728 *Figure 27* illustrates the operation of Async Mode 2 for timestamping, including the data transfer sequence.

1729 The asynchronous timing data transfer is based on the In-Band Interrupt (IBI) procedure, as described in
 1730 **Section 5.1.6.2**. Slave Devices that are capable of timestamping Async Mode 2 asynchronous events shall
 1731 have BCR [7] bit set to 1b'1. As per **Section 5.1.1.2.1**, the Master shall then read the Mandatory Data Byte
 1732 following the accepted IBI request. Bit [7] shall be set to 1b'1 to indicate that the Slave needs to transfer the
 1733 timing information of an asynchronous event; if so, then the Master shall read the following bytes and decode
 1734 the data as previously agreed between the Slave and the Master.



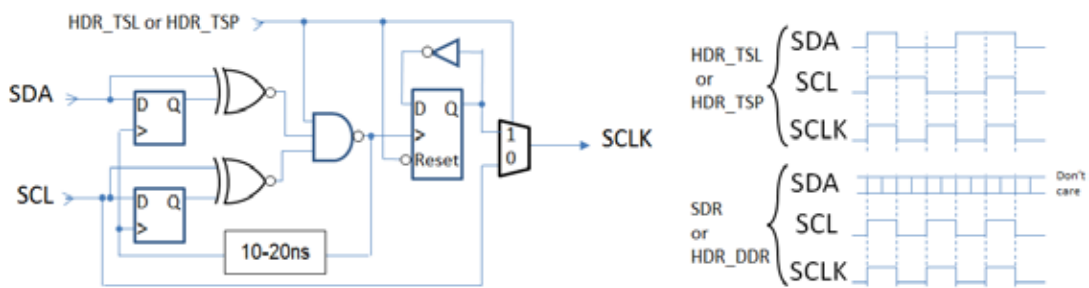
1735



1736

1737

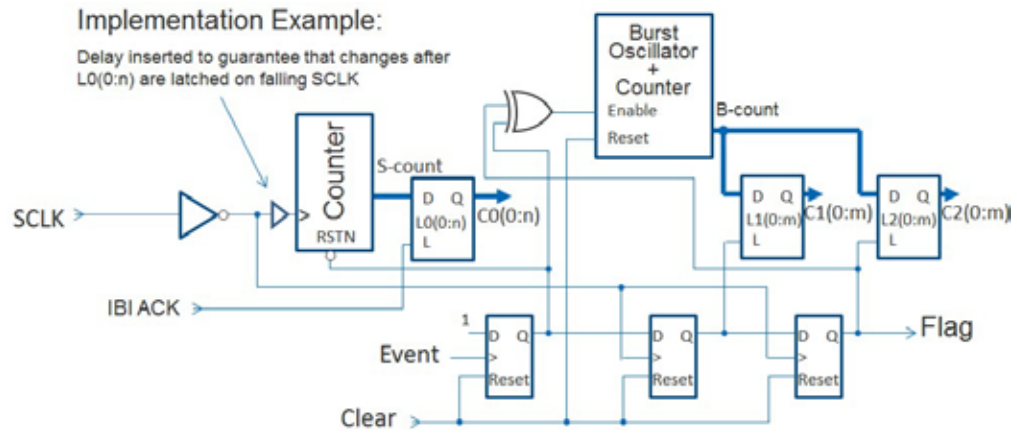
Figure 27 Example of Asynchronous Mode 2 Timestamp Data Transfer



1738

1739

Figure 28 Generating SCLK from SCL and SDA Implementation Example



1740

1741

Figure 29 Asynchronous Mode 2 Implementation Example

5.1.8.3.4 Async Mode 3: Async High-Precision Triggereable Mode

1742 Async Mode 3 has the highest accuracy and/or precision of the four Async Modes. It supports precise time
1743 triggering of multiple transducers (supporting applications such as beam forming), followed by precise time
1744 measurements of multiple sensors. Async Mode 3 is ideal for systems where higher precision and lower
1745 power consumption is needed. Like Async Mode 2, Async Mode 3 achieves higher precision by using SCL
1746 falling edges as a common coarse timing reference for Master and Slave. It achieves low power because the
1747 Slave need not generate its own clock. To achieve extended resolution, each Slave can make use of a burst
1748 oscillator.

1749 To select Async Mode 3, the Master sends the SETXTIME CCC (see *Section 5.1.9.3.20*) with the Sub-
1750 Command Enter Async Mode 3 (0xFB).

1751 The hardware implementation of Async Mode 3 in the Slave is simple; this is achieved at the expense of
1752 somewhat greater complexity on the Master side. Since there are usually many more Slaves than Masters,
1753 this is considered a reasonable trade-off. Also, Slave logic (as contrasted with Master logic) often uses a
1754 coarser technology node which, although well-suited for the higher-voltage operation that sensors often
1755 require, has a higher cost per gate, both in terms of energy and in terms of silicon area.

1756 In Async Mode 3, as in Async Mode 2, the Slave timestamp reference clock (SCLK) is derived from the I3C
1757 Bus itself (see *Figure 28*), in a manner that depends on the Bus Mode currently in use:

- 1758 · **I²C Mode, I3C SDR Mode, and I3C HDR-DDR Mode:** SCL is used as SCLK
- 1759 · **I3C HDR-TSL Mode and I3C HDR-TSP Mode:** All transitions (SDA, SCL and both
1760 simultaneously) generate SCLK

1761 Figure 31 illustrates that in operation, the Master shall initiate a sync by sending the SETXTIME CCC with
1762 the Sub-Command Async Trigger (0xFD). On the falling SCL edge during the related T-Bit, the Master and
1763 Slaves shall clear and enable their respective S-Count counters (which increment on SCLK falling edges).
1764 The fine delay settings are calibrated by a prior execution of this instruction (the burst oscillator used in the
1765 trigger is the same that is used in the timestamping operation. Therefore, the calibration that was done during
1766 the timestamp operation becomes the calibration for the Trigger. The Fine delay setting shall then have been
1767 determined by the Master after measuring the Slave's oscillator count over a given interval against the
1768 Masters own timebase. Note that if the Master sets the Slave's Coarse delay setting to 0, then the delay shall
1769 only depend upon the Fine Delay Setting. However, some variation will inevitably exist among the burst
1770 oscillators of the several Slaves, resulting in relative timing errors that increase with longer delay settings.
1771 Using Coarse in conjunction with Fine for longer delays can minimize this, but in order to have predictable
1772 delays the Master must predictably control SCL during the programmed delay interval. The Master can do
1773 so by choosing a commonly supported Mode and frequency (among the Slaves) of SCL for the delay interval
1774 required. After the transducers are pulsed, the sensors will be ready to detect events. When the sensor detects
1775 an event (see *Figure 30* and *Figure 32*), the Slave shall turn its burst oscillator on, and shall enable its B-
1776 Count counter (which shall have been previously cleared to zero). The Slave shall then initiate an IBI at the
1777 earliest opportunity (i.e., first Bus Available condition).

1778 At the first SCLK falling edge after the sensor detects the event, the value of the S-Count counter shall be
1779 loaded into register C0, and the value of the B-Count counter shall be loaded into register C1. On the second
1780 SCLK falling edge, the value of the B-Count counter shall be loaded into register C2. At this time, the burst
1781 oscillator shall be turned off, the S-Count counter shall stop, and the B-Count counter may optionally be
1782 cleared. When the IBI is acknowledged, the Slave shall transfer the contents of registers C1, C2, and C0 to
1783 the Master.

1784 The Master will have hardware similar to the Slave for generating SCLK, and shall maintains an S-Count
1785 counter that is cleared and starts counting on the same SCLK edge as the Slave. The Master's S-Count counter
1786 shall run continuously while in Async Mode 3 (following the sync pulse). The Master shall also timestamp
1787 (and data-log) its S-Count whenever a change in frequency of SCLK occurs. From the count C0 and its own
1788 data-log, the Master will be able to reconstruct the time at which C1 and C2 occurred (Tc1 and Tc2,

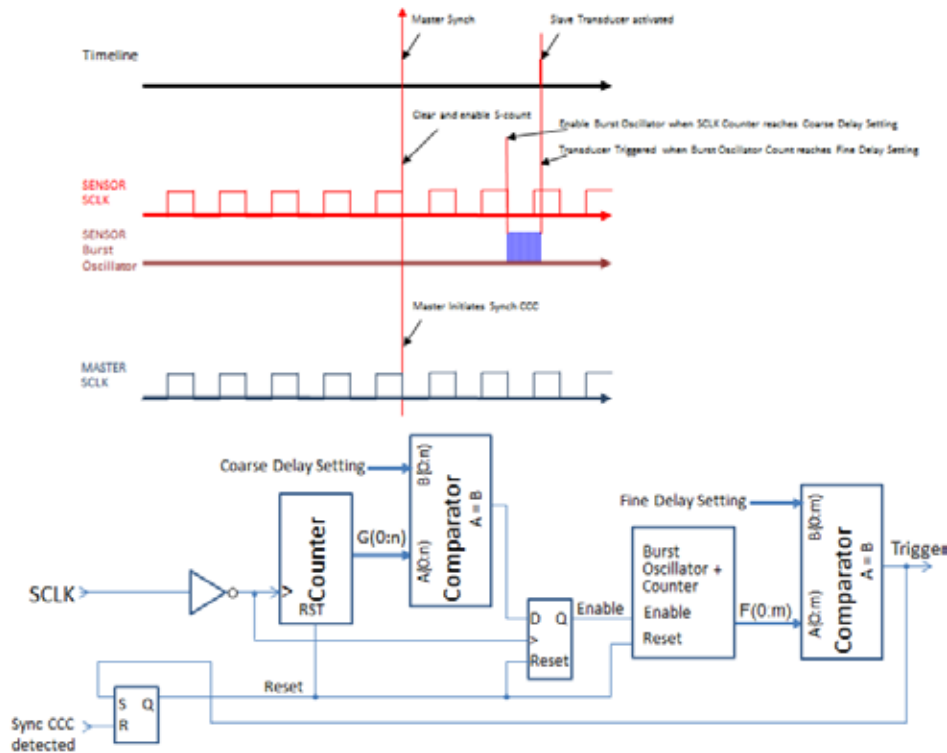
1789 respectively). The Master shall then calculate the time at which the Slave detected the event, using the
 1790 following formula:

1791
$$T_0 = T_{c1} - (T_{c2} - T_{c1}) * C_1 / (C_2 - C_1)$$

1792 Note that since the Master generates SCL (for I²C Mode, and for I3C SDR and HDR-DDR Modes), SCL is
 1793 likely to be a rational fraction of the Master’s internal timebase. This should result in a very compact and
 1794 accurate timestamp relative to this timebase. For I3C HDR-TSL and HDR-TSP Modes (see **Figure 28**),
 1795 accuracy is somewhat degraded because the Master doesn’t generate the timebase. However, high precision
 1796 among multiple sensors is maintained because the timebase is shared (as contrasted with being independently
 1797 generated in each Slave).

1798 **Figure 30** illustrates the operation of Async Mode 3 for triggering, while **Figure 31** and **Figure 32** show
 1799 timestamping, including the data transfer sequence.

1800 The asynchronous timing data transfer is based on the In-Band Interrupt (IBI) procedure, as described in
 1801 **Section 5.1.6.2**. Slave Devices that are capable of timestamping Async Mode 3 asynchronous events shall
 1802 have BCR [7] bit set to 1b’1. As per **Section 5.1.1.2.1**, the Master shall then read the Mandatory Data Byte
 1803 following the accepted IBI request, and decode the data as previously agreed between the Slave and the
 1804 Master.

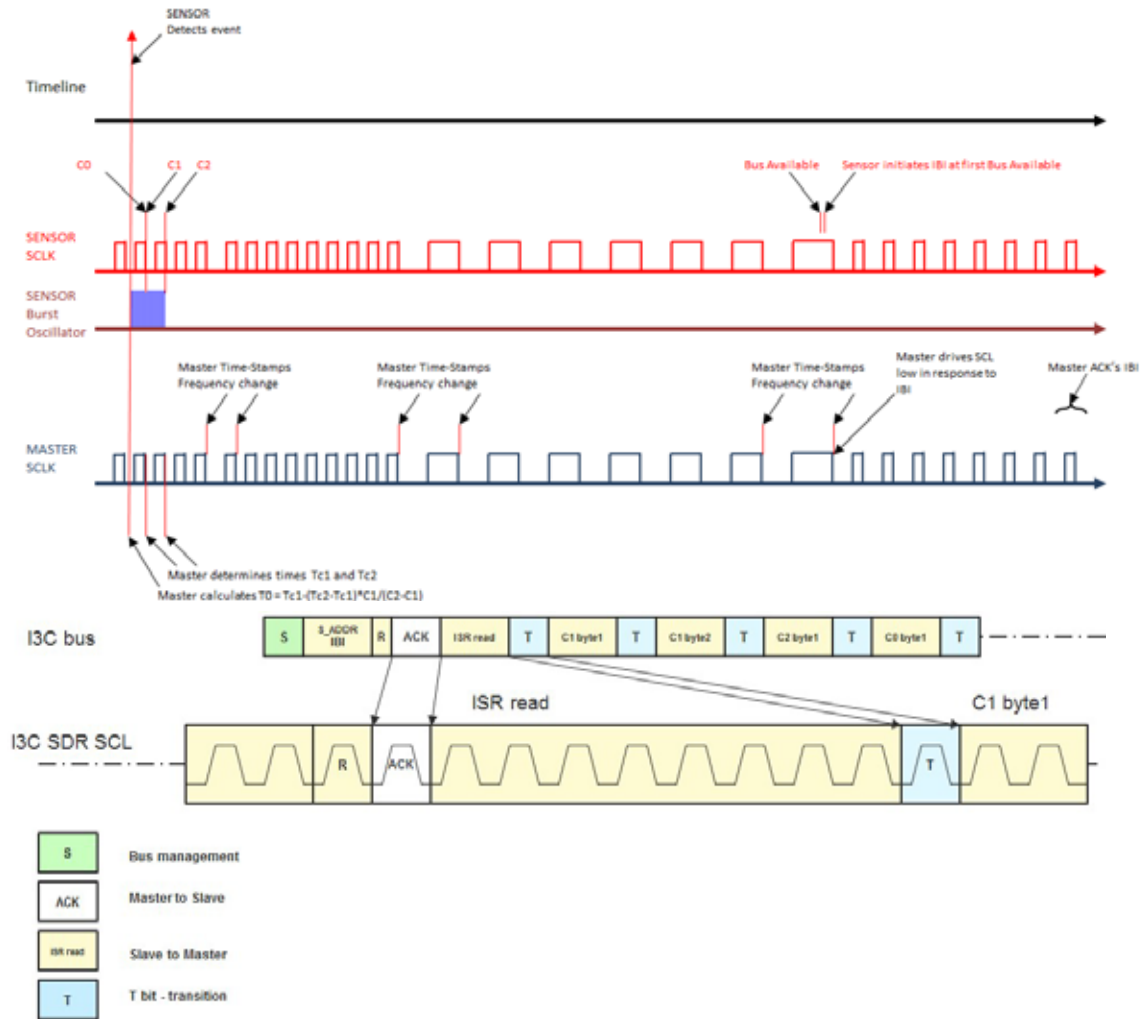


1805

1806

Figure 30 Example of Asynchronous Mode 3 Triggering

1807



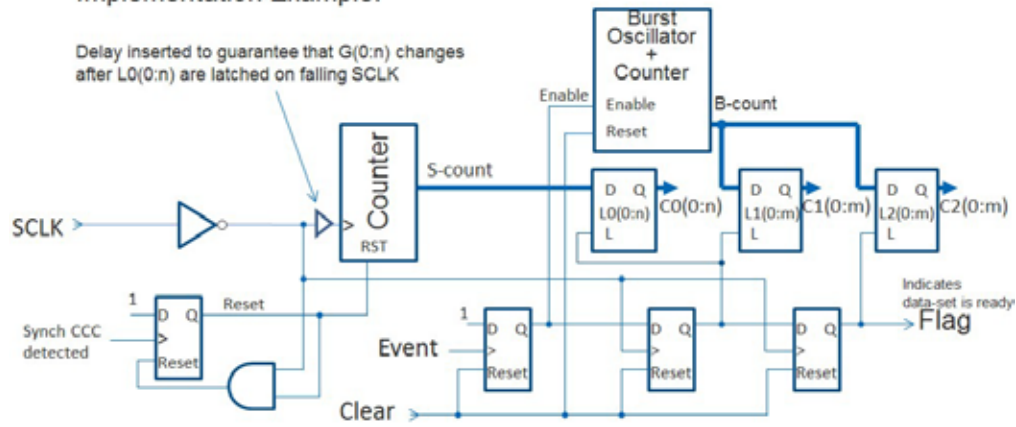
1808

1809

1810

Figure 31 Example of Asynchronous Mode 3 Timestamp Data Transfer

Implementation Example:



1811

1812

Figure 32 Asynchronous Mode 3 Implementation Example

5.1.9 Common Command Codes (CCC)

1813 Common Command Codes (CCCs) are globally supported commands that can be transmitted either directly
1814 to a specific I3C Slave Device, or to all I3C Slave Devices simultaneously. This Section specifies how CCCs
1815 are transmitted on the I3C Bus, how each CCC functions, and which CCCs I3C Devices are required to
1816 support.

5.1.9.1 CCC Command Format

1817 The CCC Command Protocol is only formatted using I3C SDR, and always starts with the I3C Broadcast
1818 Address (7'h7E). That is, after a START or Repeated START, the Address of a CCC Command shall always
1819 be a 7'h7E and the RnW bit shall always be a Write.

1820 All I3C Slaves shall recognize both the 7'h7E Broadcast Address, and their own Dynamic Address once it
1821 has been assigned. The I3C Master shall issue a CCC Command both before and after I3C Dynamic
1822 Addresses are assigned.

1823 **Note:**

1824 *Because the P_C Specification ([NXP01] Section 3.1.12) reserves the Address value 7'h7E, no*
1825 *Legacy P_C Slave will match the I3C Broadcast Address.*

1826 There are three categories of CCC Command:

- 1827 1. **Broadcast Write:** A Broadcast Write CCC is seen by all I3C Slaves. All Slaves shall inspect every
1828 received Broadcast command, even if the Slave then ignores the Broadcast command.

1829 Every Broadcast Write CCC Command ends with a Repeated START or a STOP, except
1830 ENTDAAs (see Section 5.1.9.3.4) which always ends with a STOP.

1831 **Note:**

1832 *If the CCC Command enters a Mode, then the new Mode ends according to its own rules.*

- 1833 2. **Direct Write:** A Direct Write CCC is directed to one or more specific I3C Slaves, selected by the
1834 Slave Dynamic Address(es).

1835 Every Direct Write CCC Command ends with a STOP or a Repeated START, followed by
1836 the I3C Broadcast Address (7'h7E).

- 1837 3. **Direct Read:** A Direct Read CCC reads data from one or more specific I3C Slaves, selected by the
1838 Slave Dynamic Address(es).

1839 Every Direct Read CCC Command ends with a STOP or a Repeated START, followed by
1840 the I3C Broadcast Address (7'h7E).

1841 All CCC Commands share the same general Frame format, which is shown in **Figure 33**, except for the
1842 ENTDAAs CCC (see Section 5.1.9.3.4) which always ends with a STOP. Each CCC Command has a unique
1843 Command Code. The fields within this Frame format are detailed in **Table 14**.

S	7'h7E	Command Code	Data (Optional)	Sr
Sr	/ W / ACK	/ T	(Broadcast CCC only) / T	P

1844 **Figure 33 CCC General Frame Format**

1845

Table 14 CCC Frame Field Definitions

Field	Definition	
S or Sr	A CCC always begins with either START, or Repeated START.	
7'h7E / W / ACK	This field has three parts:	
	7'h7E	The CCC Frame starts with the global Broadcast Address, so that all I3C Slaves on the Bus will see the CCC Code that follows.
	W	The Write Bit is clear (value 1'b0), indicating that the Master is writing a Message to the Slaves. This Message always includes the CCC Code, and may optionally include further Data, depending upon the value of the CCC Code.
	ACK	The collective ACK (SDA driven Low) by 1 or more I3C Slaves.
Command Code / T	An 8 bit value indicating which command is being sent, followed by a T-Bit. All defined Command Code values are specified in Section 5.1.9.3 .	
Data (Optional) / T	This field is only used with Broadcast CCC Messages, and is followed by a T-Bit. Section 5.1.9.3 specifies, for each defined CCC Code, how much Data (if any) appears here.	
Sr or P	A CCC always ends with either Repeated START, or STOP.	

5.1.9.2 Broadcast CCCs vs Direct CCCs

1846

The Command Code space is divided into Broadcast Commands and Direct Commands:

1847

- Broadcast Commands are Command Codes 0x00 to 0x7F

1848

- Direct Commands are Command Codes from 0x80 to 0xFE

1849

- Command Code 0xFF is reserved.

1850

As a result, a Slave can easily determine whether a received Command is being Broadcast to all Slaves on the I3C Bus (1'b0), or is a Direct Command (1'b1) intended only for the particular Slave, by inspecting Bit 7 (MSb) of the Command Code.

1851

1852

5.1.9.2.1 End of a CCC Command

1853

A CCC Command shall end in one of the following three I3C Bus conditions:

1854

- A STOP after the Command or Data

1855

- For a Broadcast Command, a Repeated START (for any Address value)

1856

- For a Direct Command, a Repeated START followed by 7'h7E (which may be the start of a new CCC, or may be followed by a Repeated START)

1857

1858

If the CCC Command enters a Mode, then the Mode ends according to its own rules.

1859

A 7'h7E following a Repeated START to end a Direct CCC may start another CCC, or may be followed by a Repeated START.

1860

1861

Although not a normal use, the Master may terminate a Direct CCC Command without addressing any Slave.

1862

If the Master invalidly terminates the data associated with a CCC prematurely, then the Slave shall use best efforts to handle the termination and ascertain the proper course of action. That is, the Slave may choose to disregard the event with no effect, or the Slave may process a partial impact from the incomplete data.

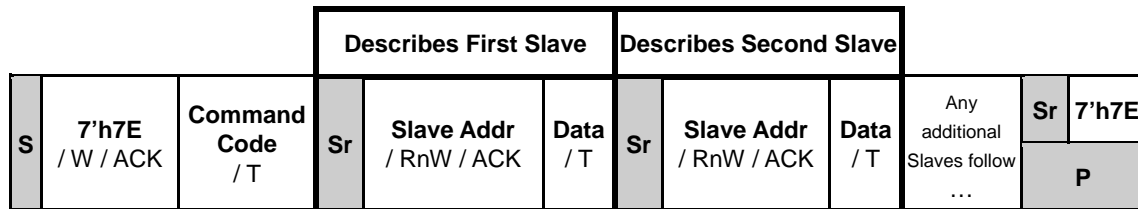
1863

1864

5.1.9.2.2 Framing Model for Direct CCC Commands

1865 In Direct CCC Commands the Frame contains first the Command Code, then one or more Repeated STARTs,
1866 then the Address of the targeted Slave Device, followed by Data, and finally either a STOP, or a Repeated
1867 START and 7'h7E.

1868 A single Direct CCC Command may also optionally address more than one Slave Device. To do this, one
1869 additional block is inserted before the final 7'h7E for each additional desired Slave Device (see *Figure 34*).
1870 Each such block consists of Repeated START, the Slave Address of the additional desired Slave Device, and
1871 the Data to be sent to that Slave Device.



1872 **Figure 34 Direct CCC Framing Model**

1873 Each addressed Slave Device may either ACK or NACK the Direct CCC Command. For a Read request, the
1874 Slave Device then returns the requested data in accordance with the SDR Standard Read model.

1875 Slaves terminate the Read in I3C, and future Direct Get CCC definitions could be extended to include
1876 additional data bytes, beyond those specified in this version of the I3C Specification. As a result, all I3C
1877 Masters Devices shall ignore any additional, unrecognized data bytes (i.e. more data bytes than this
1878 Specification defines for the given CCC Command) that the Slave Device might return in response to a Direct
1879 Get CCC.

5.1.9.2.3 Retry Model for Direct GET CCC Commands

1880 I3C mandates a single-retry model for Direct GET CCC Commands.

1881 Recall that a Direct GET CCC Command first sends an overall GET command to all Slave Devices using the
1882 Broadcast Address 7'h7E, and then sends targeted Slave Address(es) in order to stimulate per-Slave
1883 response(s). This requires the Slave to be in a state allowing an immediate response if its Address is
1884 transmitted, but also to be prepared for its Address not to be transmitted (and therefore for the Slave not to
1885 have to actually respond). For Direct GET CCC Commands that the Slave supports in hardware, such as
1886 those dealing with information locally known within the Slave (like GETBCR), this requirement generally
1887 presents no issues. But for Direct GET CCCs supported by the Slave's system, or those supported by software
1888 in the Slave, it's possible that the Slave might not be able to respond to the Direct GET CCC in time.

1889 Such situations are handled with the following single-retry model, which applies to Direct GET CCC
1890 Commands only.

1891 If a Slave cannot provide a response to a Direct GET CCC Command in time, then:

- 1892 1. The Slave shall NACK its Address.
- 1893 2. The Master shall then emit a Repeated START, followed by the Slave Address. Note that this is a
1894 second transmission of the Slave Address. This gives the Slave extra time to prepare its response.
1895 As a further measure to give the Slave even more time to prepare its response, the Master may
1896 optionally also employ a clock delay (as per *Section 5.1.2.5*) after the Slave's NACK and before
1897 the Master's Repeated START.
- 1898 3. The Slave should respond to the retry attempt in step 2 with an ACK, and then transmit the results
1899 requested by the Direct GET CCC.

1900 This Retry Model is limited to a single retry. Any Slave still unable (for any reason) to respond to the retry
1901 attempt from step 2 will NACK it. If a Slave NACKs the second attempt in step 2, then the Master shall not
1902 attempt further retries to that Slave.

1903 Note that step 2 only occurs if the Slave NACKs the original Direct GET CCC request. As a result, this retry
1904 model never imposes a time penalty, except in cases where the Slave actually needs the extra time.

5.1.9.3 CCC Command Definitions

1905 Every I3C Common Command Code (CCC) marked as 'Required' in **Table 15** shall be supported by all I3C
 1906 Master Devices and by all I3C Slave Devices. When a non-'Required' CCC is supported by an I3C Device,
 1907 it shall be implemented as defined in this Specification.

1908

Table 15 I3C Common Command Codes

Command Code	CCC Type	Required	Command Name	Default	Section	Brief Description
0x00	Broadcast	Y	ENEC Enable Events Command	Y	5.1.9.3.1	Enable Slave event driven interrupts
0x01	Broadcast	Y	DISEC Disable Events Command	N	5.1.9.3.1	Disable Slave event driven interrupts
0x02	Broadcast	Y ¹	ENTAS0 Enter Activity State 0	Y	5.1.9.3.2	Set Activity Mode to State 0 (normal operation)
0x03	Broadcast	N ¹	ENTAS1 Enter Activity State 1	N	5.1.9.3.2	Set Activity State 1
0x04	Broadcast	N ¹	ENTAS2 Enter Activity State 2	N	5.1.9.3.2	Set Activity State 2
0x05	Broadcast	N ¹	ENTAS3 Enter Activity State 3	N	5.1.9.3.2	Set Activity State 3
0x06	Broadcast	Y	RSTDAA Reset Dynamic Address Assignment	–	5.1.9.3.3	Forget current Dynamic Address and wait for new assignment
0x07	Broadcast	Y	ENTDAA Enter Dynamic Address Assignment	–	5.1.9.3.4	Entering Master initiation of Slave Dynamic Address Assignment. Don't participate if the Slave already has an Address assigned.
0x08	Broadcast	N	DEFSLVS Define List of Slaves	–	5.1.9.3.7	Master defines Dynamic Address, DCR Type, and Static Address (or 0) per Slave
0x09	Broadcast	Y ⁶	SETMWL Set Max Write Length	–	5.1.9.3.5	Maximum write length in a single command
0x0A	Broadcast	Y ⁷	SETMRL Set Max Read Length	–	5.1.9.3.6	Maximum read length in a single command
0x0B	Broadcast	N	ENTTM Enter Test Mode	–	5.1.9.3.8	Master has entered Test Mode
0x0C – 0x1F	–	–	<i>MIPI Reserved</i>	–	–	<i>Reserved for future use by MIPI Alliance</i>
0x20	Broadcast	N ³	ENTHDR0 Enter HDR Mode 0	–	5.1.9.3.9	Master has entered HDR – DDR Mode
0x21	Broadcast	N ³	ENTHDR1 Enter HDR Mode 1	–	5.1.9.3.9	Master has entered HDR – TSP Mode
0x22	Broadcast	N ³	ENTHDR2 Enter HDR Mode 2	–	5.1.9.3.9	Master has entered HDR – TSL Mode
0x23	Broadcast	N ³	ENTHDR3 Enter HDR Mode 3	–	–	<i>Master has entered HDR - Future</i>

Command Code	CCC Type	Required	Command Name	Default	Section	Brief Description
0x24	Broadcast	N ³	ENTHDR4 Enter HDR Mode 4	–	–	Master has entered HDR - Future
0x25	Broadcast	N ³	ENTHDR5 Enter HDR Mode 5	–	–	Master has entered HDR - Future
0x26	Broadcast	N ³	ENTHDR6 Enter HDR Mode 6	–	–	Master has entered HDR - Future
0x27	Broadcast	N ³	ENTHDR7 Enter HDR Mode 7	–	–	Master has entered HDR - Future
0x28	Broadcast	N	SETXTIME Exchange Timing Information	–	5.1.9.3.20	Framework for exchanging event timing information
<i>0x29 – 0x48</i>	–	–	<i>MIPI Sensor WG Reserved</i>	–	–	<i>Reserved for future use by MIPI Sensor WG</i>
<i>0x49 – 0x57</i>	Broadcast	–	<i>MIPI Reserved for other WG's – Broadcast CCCs</i>	–	–	<i>Reserved for future use by other MIPI Working Groups</i>
<i>0x58 – 0x5B</i>	Broadcast	–	<i>MIPI Debug WG Reserved – Broadcast CCCs</i>	–	–	<i>Reserved for use by MIPI Debug Working Group</i>
<i>0x5C – 0x60</i>	Broadcast	–	<i>MIPI RIO WG Reserved – Broadcast CCCs</i>	–	–	<i>Reserved for use by MIPI RIO Working Group</i>
0x61 – 0x7F	Broadcast	–	Vendor Extension – Broadcast CCCs	–	–	For Vendor use
0x80	Direct	Y	ENEC Enable Events Command	Y	5.1.9.3.1	Enable Slave event driven interrupts
0x81	Direct	Y	DISEC Disable Events Command	N	5.1.9.3.1	Disable Slave event driven interrupts
0x82	Direct	Y ¹	ENTAS0 Enter Activity State 0	Y	5.1.9.3.2	Set Activity Mode to State 0 (normal operation)
0x83	Direct	N ¹	ENTAS1 Enter Activity State 1	N	5.1.9.3.2	Set activity State 1
0x84	Direct	N ¹	ENTAS2 Enter Activity State 2	N	5.1.9.3.2	Set activity State 2
0x85	Direct	N ¹	ENTAS3 Enter Activity State 3	N	5.1.9.3.2	Set activity State 3
0x86	Direct	Y	RSTDAA Reset Dynamic Address Assignment	–	5.1.9.3.3	Forget current Dynamic Address and wait for new assignment
0x87	Direct Set	N	SETDASA Set Dynamic Address from Static Address	–	5.1.9.3.10	Master assigns a Dynamic Address to a Slave with a known Static Address.
0x88	Direct Set	Y	SETNEWDA Set New Dynamic Address	–	5.1.9.3.11	Master assigns a new Dynamic Address to any I3C Slave

Command Code	CCC Type	Required	Command Name	Default	Section	Brief Description
0x89	Direct Set	Y ²	SETMWL Set Max Write Length	–	5.1.9.3.5	Maximum write length in a single command
0x8A	Direct Set	Y ²	SETMRL Set Max Read Length	–	5.1.9.3.6	Maximum read length in a single command
0x8B	Direct Get	Y ²	GETMWL Get Max Write Length	–	5.1.9.3.5	Get Slave's maximum possible write length
0x8C	Direct Get	Y ²	GETMRL Get Max Read Length	–	5.1.9.3.6	Get a Slave's maximum possible read length
0x8D	Direct Get	Y	GETPID Get Provisional ID	–	5.1.9.3.12	Get a Slave's Provisional ID
0x8E	Direct Get	Y	GETBCR Get Bus Characteristics Register	–	5.1.9.3.13	Get a Device's Bus Characteristic Register (BCR)
0x8F	Direct Get	Y	GETDCR Get Device Characteristics Register	–	5.1.9.3.14	Get a Device's Device Characteristics Register (DCR)
0x90	Direct Get	Y	GETSTATUS Get Device Status	–	5.1.9.3.15	Get a Device's operating status
0x91	Direct Get	N	GETACCMST Get Accept Mastership	–	5.1.9.3.16	Current Master is requesting and confirming a Bus Mastership from a Secondary Master
0x92	–	–	<i>MIPI Sensor WG Reserved</i>	–	–	<i>Reserved for future use by MIPI Sensor WG</i>
0x93	Direct Set	N	SETBRGTGT Set Bridge Targets	–	5.1.9.3.17	Master tells Bridge (to/from I ² C, SPI, UART, etc.) what endpoints it is talking to (by Dynamic Address and type/ID).
0x94	Direct Get	N ⁴	GETMXDS Get Max Data Speed	–	5.1.9.3.18	Master asks Slave for its SDR Mode max. Read and Write data speeds (& optionally max. Read Turnaround time)
0x95	Direct Get	N ⁵	GETHRCAP Get HDR Capability	–	5.1.9.3.19	Master asks Slave what HDR Modes it supports
0x96 – 0x97	–	–	<i>MIPI Sensor WG Reserved</i>	–	–	<i>Reserved for future use by MIPI Sensor WG</i>
0x98	Direct	N	SETXTIME Set Exchange Timing Information	–	5.1.9.3.20	Framework for exchanging event timing information
0x99	Direct	N	GETXTIME Get Exchange Timing Information	–	5.1.9.3.21	Framework for exchanging event timing information
0x9A – 0xBF	<i>Direct</i>	–	<i>MIPI Sensor WG Reserved – Direct CCCs</i>	–	–	<i>Reserved for future use by MIPI Sensor WG</i>

Command Code	CCC Type	Required	Command Name	Default	Section	Brief Description
0xC0 – 0xD6	Direct	–	MIPI Reserved for other WG's – Direct CCCs	–	–	Reserved for future use by MIPI Sensor WG
0xD7 – 0xDA	Direct	–	MIPI Debug WG Reserved – Direct CCCs	–	–	Reserved for use by MIPI Debug WG
0xDB – 0xDF	Direct	–	MIPI RIO WG Reserved – Direct CCCs	–	–	Reserved for use by MIPI RIO WG
0xE0 – 0xFE	Direct	–	Vendor Extension – Direct CCCs	–	–	For Vendor use
0xFF	–	–	MIPI Sensor WG Reserved	–	–	Reserved for future use by MIPI Sensor WG

Note:

- 1) Slave Devices shall be permitted to self-power-manage based on this information.
- 2) This CCC shall be supported by Devices capable of transporting 16 or more sequential bytes.
- 3) All I3C compliant Slaves shall recognize that the Bus is entering an HDR Mode, and detect the HDR Exit Pattern.
- 4) This CCC shall be supported by Slave Devices with Bus Control Register (BCR) Bit [0] set to 1'b1.
- 5) This CCC shall be supported by Slave Devices that support any HDR Mode.
- 6) See **Section 5.1.9.3.5 Set/Get Max Write Length (SETMWL/GETMWL)**.
- 7) See **Section 5.1.9.3.6 Set/Get Max Read Length (SETMRL/GETMRL)**.

5.1.9.3.1 Enable/Disable Slave Events Command (ENEC/DISEC)

1909 These four Direct (*Table 16*) or Broadcast (*Table 17*) CCCs allows the Master to control when Slave-initiated
 1910 traffic is (Enable) vs. is not (Disable) allowed on the I3C Bus. This control governs a Slave's attempts to
 1911 request an Interrupt (ENINT/DISINT), to request Mastership (ENMR/DISMR), or to signify a Hot-Join event
 1912 (ENHJ/DISHJ).

1913 **Table 16 ENEC/DISEC Format 1: Direct**

S	7'h7E / W / ACK	Direct ENEC/DISEC CCC / T	Sr	Slave Addr / W / ACK	En/Dis Slave Event Byte / T	Sr	7'h7E
						P	

1914 **Table 17 ENEC/DISEC Format 2: Broadcast**

S	7'h7E / W / ACK	Broadcast ENEC/DISEC CCC / T	Enable/Disable Slave Event Byte / T	Sr	7'h7E
Sr				P	

1915 *Table 18* and *Table 19* show the bits to set in the Slave Event Byte to enable and disable, respectively, the
 1916 four supported I3C Slave/Secondary Master event types. Reserved bits are for future MIPI use.

1917 **Table 18 Enable Slave Events Command Byte Format**

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reserved				ENHJ	Reserved	ENMR	ENINT

1918 **Table 19 Disable Slave Events Command Byte Format**

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reserved				DISHJ	Reserved	DISMR	DISINT

1919 The supported I3C Slave/Secondary Master event types are:

- 1920 • **Slave Interrupt Requests:** Enable (ENINT) / Disable (DISINT)
 1921 These bits allow the Master to control when Slave-initiated Interrupts are (ENINT) and are not
 1922 (DISINT) allowed on the I3C Bus.
- 1923 • **Mastership Requests:** Enable (ENMR) / Disable (DISMR)
 1924 These bits allow the Current Master to control when Mastership requests from Secondary Masters
 1925 are (ENMR) and are not (DISMR) allowed on the I3C Bus.
- 1926 • **Hot-Join Event:** Enable (ENHJ) / Disable (DISHJ)
 1927 These bits allow the Master to control when Slave-initiated Hot-Join is (ENHJ) and is not (DISHJ)
 1928 allowed on the I3C Bus. The Master can Broadcast this CCC in order to command Devices to
 1929 refrain from making Dynamic Address Assignment requests until later authorized by the Master, in
 1930 case the Master isn't ready to service the Hot-Joining Device(s). (Hot-Join events are
 1931 asynchronous.)

5.1.9.3.2 Enter Activity State 0–3 (ENTAS0–ENTAS3)

1932 These four Direct (*Table 20*) and four Broadcast (*Table 21*) CCCs allow the Master to inform one or all Slave
1933 Devices that it will not be active on the I3C Bus for an approximated amount of time, so that the Slave
1934 Devices may use a lower power state during that period. There is one Direct CCC and one Broadcast CCC
1935 per Activity State. All I3C Slave Devices shall maintain I3C communications capabilities in all Activity
1936 States.

1937 **Table 20 ENTASx Format 1: Direct**

S	7'h7E /W / ACK	Direct ENTASx CCC /T	Sr	Slave Addr /W / ACK	Sr	7'h7E
					P	

1938 **Table 21 ENTASx Format 2: Broadcast**

S	7'h7E /W / ACK	Broadcast ENTASx CCC /T	Sr	7'h7E
Sr			P	

1939 *Table 22* below gives the expected minimum Bus activity interval for each Activity State.

1940 **Table 22 Enter Activity State CCCs (ENTASx)**

CCC	Activity State	Minimum Bus Activity Interval
ENTAS0	Activity State 0	1 μSec: Latency-free operation
ENTAS1	Activity State 1	100 μSec
ENTAS2	Activity State 2	2 mSec
ENTAS3	Activity State 3	50 mSec: Lowest-activity operation

5.1.9.3.3 Reset Dynamic Address Assignment (RSTDAA)

1941 This Direct (*Table 23*) or Broadcast (*Table 24*) CCC indicates to one or all I3C Devices that the Master
1942 requires them to clear/reset their Master-assigned Dynamic Address. After clearing their Dynamic Address,
1943 the Devices are ready to participate in a Dynamic Address Assignment procedure, per *Section 5.1.4*.

1944 **Table 23 RSTDAA Format 1: Direct**

S	7'h7E /W / ACK	Direct RSTDAA CCC /T	Sr	Slave Addr /W / ACK	Sr	7'h7E
					P	

1945 **Table 24 RSTDAA Format 2: Broadcast**

S	7'h7E /W / ACK	Broadcast RSTDAA CCC /T	Sr	7'h7E
Sr			P	

5.1.9.3.4 Enter Dynamic Address Assignment (ENTDAA)

1946 This Broadcast CCC (*Table 25*) indicates to all I3C Devices that the Master requires them to enter the
 1947 Dynamic Address Assignment procedure described in *Section 5.1.4*. Slave Devices that already have a
 1948 Dynamic Address assigned shall not respond to this command.

1949 To exit from ENTDAA mode, the Master shall issue a STOP, and shall not use a Repeated START.

1950

Table 25 ENTDAA Format

S	7'h7E / W / ACK	ENTDAA CCC / T	P
Sr			

5.1.9.3.5 Set/Get Max Write Length (SETMWL/GETMWL)

1951 These Direct and Broadcast CCCs (Direct Set or Get in *Table 26*, Broadcast Set in *Table 27*) allow the I3C
 1952 Master to Set or Get a maximum data write length in bytes for one Slave Device. This Max Write Length
 1953 does not affect data write lengths for Broadcast CCCs. The Set/Get Max Write Length value is transmitted
 1954 over two bytes, with the most significant byte (MSb) transmitted first. The minimum value that Max Write
 1955 Length can be set to is 8.

1956 This CCC is required if (and only if) any private Write Message(s) and/or any extended Write CCC(s)
 1957 implemented by the Slave Device support a variable limit on the maximum number of data bytes per
 1958 Message, and this limit is greater than 8 bytes. This allows the Slave to limit the number of bytes the Master
 1959 sends. A Slave Device with no such settable limit may optionally support this CCC, but is not expected to do
 1960 so.

1961

Table 26 Direct SETMWL/GETMWL Format

S	7'h7E / W / ACK	SETMWL or GETMWL CCC / T	Sr	Slave Addr / RnW / ACK	MWL MSb / T	MWL LSb / T	Sr	7'h7E
							P	

1962

Table 27 Broadcast SETMWL Format

S	7'h7E / W / ACK	SETMWL CCC / T	MWL MSb / T	MWL LSb / T	Sr	7'h7E
					P	

5.1.9.3.6 Set/Get Max Read Length (SETMRL/GETMRL)

1963 These Direct and Broadcast CCCs (Direct Set or Get in *Table 28*, Broadcast Set in *Table 29*) allow the I3C
1964 Master to Set or Get a maximum data read length, and optionally a maximum IBI payload size.

1965 The Set/Get Max Read Length value is transmitted over the first two bytes, with most significant byte (MSb)
1966 transmitted first. The minimum value to which Max Read Length can be set is 16.

1967 For devices with BCR bit 2 set to 1'b1, the Max IBI payload size value is added as a third byte, where a value
1968 of 0 indicates an unlimited payload size. If Timing Control is used, then the minimum IBI payload size is
1969 either four bytes or five bytes. If Timing Control (see *Section 5.1.8*) is not used, then the minimum IBI
1970 payload size is one (one byte).

1971 This CCC is optional for the Slave, with two exceptions:

- 1972 1. This CCC is required if both (a) any private Read Request Message(s) and/or any extended Read
1973 Request CCC(s) implemented by the Slave support a variable limit on the maximum number of
1974 data bytes that the Slave may return per Message, and (b) this limit is greater than 16 bytes.
- 1975 2. This CCC is required if the Slave both (a) supports an IBI Payload (as indicated with BCR bit 1),
1976 and (b) will transmit more than one byte of private payload (not counting Timing Control bytes,
1977 when Timing Control used).

1978 **Table 28 Direct SETMRL/GETMRL Format**

S	7'h7E /W/ACK	SETMRL or GETMRL CCC /T	Sr	Slave Addr /RnW/ACK	MRL MSb /T	MRL LSb /T	IBI Payload Size /T	Sr	7'h7E
								P	

1979 **Table 29 Broadcast SETMRL Format**

S	7'h7E /W/ACK	SETMRL CCC /T	MRL MSb /T	MRL LSb /T	IBI Payload Size /T	Sr	7'h7E
						P	

5.1.9.3.7 Define List of Slaves (DEFSLVS)

1980 This Broadcast CCC (*Table 30*) is only relevant to Secondary Masters, which may independently elect either
1981 to respond to this CCC, or to ignore it. This CCC tells Secondary Master Devices what Slaves are present on
1982 the I3C Bus, via four consecutive Data Bytes per Slave. First the Current Master identifies itself by
1983 transmitting its own data in the first set of four data bytes, using the value 7'h7E as the Static Address. Then
1984 each additional Slave on the I3C Bus Slave is represented by a further set of four data bytes.

1985 **Table 30 DEFSLVS Format**

				Describes Current Master				Describes First Slave (Any additional Slaves will follow)					
S	7'h7E /W /ACK	DEFSLVS CCC /T	Count /T	Dynamic Addr Master /T	DCR Type Master /T	BCR Type Master /T	Static Addr 7'h7E /T	Dynamic Addr 0 /T	DCR Type 0 /T	BCR Type 0 /T	Static Addr 0 /T	Sr	7'h7E
Sr												P	

1986 Count is the number of Slaves present on the I3C Bus. Each Slave is represented by a set of four data bytes:

- 1987 1. **Dynamic Address:** The 7 most significant bits (Bits [7:1]) contain the current value of the Slave's
1988 Main Master-assigned 7-bit Dynamic Address, and the least significant bit (Bit [0]) is filled with
1989 the value 1'b0. For Legacy I²C Devices, the value of the Dynamic Address shall be 7'h00.
- 1990 2. **DCR Type:** The Slave's Device Characteristics Register value, or 0x00 if unknown. For Legacy
1991 I2C Devices the value of the DCR Type shall be the Device's LVR value.

- 1992 3. **BCR Type:** The Slave's Bus Characteristics Register value, or 0x00 if unknown.
- 1993 4. **Static Address:** The Slave's original 7-bit static I²C Address in the 7 most significant bits
- 1994 (Bits [7:1]) with the least significant bit (Bit [0]) filled with the value 1'b0. If no Static Address,
- 1995 then the value shall be 7'h00.
- 1996 For a Legacy I²C Device, the value of the Dynamic Address field will be 7'h00, and the DCR field will
- 1997 contain the value of the Device's Legacy Virtual Register (LVR).
- 1998 The Master shall not send the DEFSLVS CCC unless at least one Secondary Master Device is present on the
- 1999 I3C Bus. The Master shall send the DEFSLVS CCC following every "Hot-Join" event.

5.1.9.3.8 Enter Test Mode (ENTTM)

2000 This Broadcast CCC (*Table 31*) informs all I3C Devices that the Master is entering a specified Test Mode

2001 during manufacturing or Device test. The Enter Test Mode command Frame format includes a byte that

2002 specifies which Test Mode to enter. Supporting I3C Devices shall enter the indicated Test Mode upon receipt

2003 of the Enter Test Mode CCC. *Table 32* lists the defined Test Mode byte values.

2004

Table 31 ENTTM Format

S	7'h7E / W / ACK	ENTTM CCC / T	Test Mode Byte / T	Sr	7'h7E
					P

2005

Table 32 ENTTM Test Mode Byte Values

Byte Value	I3C Test Mode	Description
0x00	Exit Test Mode	This value removes all I3C Devices from Test Mode
0x01	Vendor Test Mode	This value indicates that I3C Devices shall return a random 32-bit value in the Provisional ID during the Dynamic Address Assignment procedure
0x02 – 0xFF	MIPI Reserved	Reserved for future use by the MIPI Alliance

5.1.9.3.9 Enter HDR Mode 0–7 (ENTHDR0–ENTHDR7)

2006 These eight Broadcast CCCs inform all I3C Devices that the Bus is being switched into the indicated HDR

2007 Mode. See the indicated Section of this Specification for further details on each HDR Mode.

2008

Table 33 Enter HDR Mode CCCs (ENTHDRx)

CCC	Enter HDR Mode	Mode Name	See Section
ENTHDR0	HDR Mode 0	HDR-DDR	Section 5.2.2
ENTHDR1	HDR Mode 1	HDR-TSP	Section 5.2.3
ENTHDR2	HDR Mode 2	HDR-TSL	Section 5.2.3
ENTHDR3 to ENTHDR7	HDR Modes 3-7	Reserved for future definition by MIPI Alliance	

5.1.9.3.10 Set Dynamic Address from Static Address (SETDASA)

2009 This CCC (*Table 34* and *Table 35* illustrate the two distinct command formats) allows the Master to assign a
2010 Dynamic Address to one Slave using the Slave’s Static Address. This is faster than the ENTDAAs Dynamic
2011 Address Assignment procedure (see *Section 5.1.9.3.4*). The SETDASA CCC should be used before the
2012 ENTDAAs CCC is used; all Slaves without assigned Dynamic Addresses will respond to the ENTDAAs CCC.

2013 **Note:**

2014 *The SETDASA Command Code is unusual in that it addresses the desired I3C Device by its I²C*
2015 *Static Address, rather than by its Dynamic Address. As a result, this CCC can only be sent to a Slave*
2016 *that has an I²C Static Address.*

2017 The newly assigned Address is transmitted in the Dynamic Address Byte shown in *Table 34*, where the 7
2018 most significant bits (Bits [7:1]) contain the 7-bit Dynamic Address, and the least significant bit (Bit [0]) is
2019 filled with the value 1’b0.

2020 **Table 34 SETDASA Format 1: Primary**

S	7’h7E / W / ACK	SETDASA CCC / T	Sr	Slave Addr / W / ACK	7-bit Dynamic Address / 0 / T	Sr	7’h7E
						P	

SETDASA Minimal Bus Point-to-Point Communication

2021 The SETDASA CCC may also be specially used for simple point-to-point communication in I3C Minimal
2022 Bus use cases. An I3C Minimal Bus is an I3C Bus with one I3C Master Device (potentially with reduced
2023 functionality), and one I3C Slave Device. In this special usage of the SETDASA CCC, the Master Device
2024 both uses the fixed value 7’h01 as the Static Address, and uses the fixed (and reserved) value of 7’h01 as the
2025 Dynamic Address (see *Table 35*). This special usage of the SETDASA CCC allows for simpler Master
2026 Devices, and optionally simpler Slave Devices intended for use specifically in Minimal Bus configurations.

2027 I3C Slave Devices should support this special usage of the SETDASA CCC unless such support would make
2028 the Slave Device unusable in a Minimal Bus use case. A Slave Device supporting this special Minimal Bus
2029 usage of the SETDASA CCC shall match the Static Address 7’h01, and then accept 7’h01 as its new Dynamic
2030 Address (same as the natural result of SETDASA). The Slave Device may choose to behave differently after
2031 receiving its Dynamic Address in this manner.

2032 An I3C Master shall not use this special form of the SETDASA CCC unless the I3C Bus is known to have
2033 precisely one Master and either:

- 2034 · One active I3C Slave Device that is capable of supporting this special usage of the SETDASA
2035 CCC, or
- 2036 · One or more I3C Slave Devices that act strictly as receivers, i.e., that do not use In-Band Interrupt
2037 and that will not be sent Read requests. This arrangement permits a single Master Device to, in
2038 effect, Broadcast to the Slave Devices.

2039 **Table 35 SETDASA Format 2: Point-to-Point**

S	7’h7E / W / ACK	SETDASA CCC / T	Sr	7’h01 / W / ACK	7’h01 / 0 / T	Sr	7’h7E
						P	

5.1.9.3.11 Set New Dynamic Address (SETNEWDA)

2040 This Direct CCC (*Table 36*) allows the I3C Master to assign a new Dynamic Address to one I3C Slave
 2041 Device. In the Dynamic Address field, the 7 most significant bits (Bits [7:1]) contain the 7-bit Dynamic
 2042 Address, and the least significant bit (Bit [0]) is filled with the value 1'b0.

2043

Table 36 SETNEWDA Format

S	7'h7E /W/ACK	SETNEWDA CCC /T	Sr	Slave Addr /W/ACK	7-bit Dynamic Address / 0 /T	Sr	7'h7E
						P	

5.1.9.3.12 Get Provisional ID (GETPID)

2044 This Direct CCC (see *Figure 19* and *Table 37*) is a Get request for one I3C Slave Device to return its 48-bit
 2045 Provisional ID to the Master, as described in *Section 5.1.4*. Following transmission of the GETPID CCC, the
 2046 48-bit value is transmitted as 6 bytes, with MSb first.

2047

Table 37 GETPID Format

S	7'h7E /W/ACK	GETPID CCC /T	Sr	Slave Addr /R/ACK	GETPID	GETPID	GETPID	GETPID	GETPID	GETPID	Sr	7'h7E
					Byte 5 /T	Byte 4 /T	Byte 3 /T	Byte 2 /T	Byte 1 /T	Byte 0 /T		

5.1.9.3.13 Get Bus Characteristics Register (GETBCR)

2048 This Direct CCC (*Table 38*) is a Get request for one I3C Slave Device to return its Bus Characteristics
 2049 Register (BCR) to the Master, as described in *Section 5.1.1.2.1*. The BCR value is transmitted in one byte,
 2050 with the MSb transmitted first.

2051

Table 38 GETBCR Format

S	7'h7E /W/ACK	GETBCR CCC /T	Sr	Slave Addr /R/ACK	GETBCR Byte /T	Sr	7'h7E
						P	

5.1.9.3.14 Get Device Characteristics Register (GETDCR)

2052 This Direct CCC (*Table 39*) is a Get request for one I3C Slave Device to return its Device Characteristics
 2053 Register (DCR) to the Master, as described in *Section 5.1.1.2.2*. The DCR value is transmitted in one byte,
 2054 with the MSb transmitted first.

2055

Table 39 GETDCR Format

S	7'h7E /W/ACK	GETDCR CCC /T	Sr	Slave Addr /R/ACK	GETDCR Byte /T	Sr	7'h7E
						P	

5.1.9.3.15 Get Device Status (GETSTATUS)

2056 This Direct CCC (*Table 40*) is a Get request for one I3C Slave Device to return its current Status, in the two-
2057 byte format detailed in *Table 41*. Note that byte 0 is the LSb, and byte 1 is the MSb.

2058

Table 40 GETSTATUS Format

S	7'h7E /W / ACK	GETSTATUS CCC / T	Sr	Slave Addr / R / ACK	GETSTATUS MSb / T	GETSTATUS LSb / T	Sr	7'h7E
								P

2059

Table 41 GETSTATUS MSb-LSb Format

Bits	Field	Description
15:8	Vendor Reserved	Reserved for vendor-specific meaning.
7:6	Activity Mode	Contains the two-bit ID of the Slave Device's current Activity Mode (readiness to support data read of sensor or related information).
5	Protocol Error	If set to 1'b1, then the Slave detected a protocol error since the last Status read. The Slave might or might not be able to check for such errors. Note that this value self-clears upon every successful completion of a Master read of the Slave's Status.
4	Reserved	Reserved for future definition by the MIPI Sensor WG.
3:0	Pending Interrupt	Contains the interrupt number of any pending interrupt, or 0 if no interrupts are pending. This encoding allows for up to 15 numbered interrupts. If more than one interrupt is set, then the highest priority interrupt shall be returned.

5.1.9.3.16 Get Accept Mastership (GETACCMST)

2060 This Direct Get CCC (*Table 42*) is used both to verify a Master Request, and to allow the Current Master to
 2061 offer Mastership to an I3C Secondary Master. The Get is used to confirm acceptance; to do so, the Secondary
 2062 Master returns its 7-bit Dynamic Address as shown in *Table 42*. The value of the 7-bit Dynamic Address will
 2063 be the same as the value of the Slave Address. If the Secondary Master does not accept, then it shall NACK
 2064 the Get request as shown in *Table 43*.

2065 A Secondary Master requesting Mastership will gain Mastership only if all four of the following steps occur
 2066 in the indicated order. For a Current Master offering Mastership, only steps 2 through 4 apply.

- 2067 1. The Current Master ACKs the Mastership request
- 2068 2. The Current Master transmits a GETACCMST command to the Secondary Master
- 2069 3. The Secondary Master correctly replies with its 7-bit Dynamic Address. The value of the 7-bit
 2070 Dynamic Address will be the same as the value of the Slave Address.

2071 If the Secondary Master instead responds with NACK (*Table 43*), or with an incorrect 7-
 2072 bit Address (*Table 44*), then:

- 2073 · The Secondary Master will not acquire Mastership
 - 2074 · The GETACCMST command is canceled, and
 - 2075 · The Current Master can then either (a) Retry the CCC, or (b) End the transaction by
 2076 providing a STOP.
- 2077 4. The Current Master issues a STOP

2078 If the Current Master instead issues a Repeated START, then:

- 2079 · The Secondary Master will not gain Mastership
- 2080 · The GETACCMST command is canceled (*Table 44*), and
- 2081 · The Current Master can then either (a) Retry the CCC, or (b) End the transaction by
 2082 providing a STOP.

2083 **Table 42 GETACCMST Format 1: Accepted**

S	7'h7E / W / ACK	GETACCMST CCC / T	Sr	Slave Addr / R / ACK	7-Bit Dynamic Address / PAR / T	P
---	--------------------	-------------------------	----	-------------------------	---------------------------------------	---

2084 **Note:**

2085 *The value of the 7-Bit Dynamic Address will be the same as the value of the Slave Addr.*

2086 **Table 43 GETACCMST Format 2: Not Accepted**

S	7'h7E / W / ACK	GETACCMST CCC / T	Sr	Slave Addr / R / NACK	Sr	7'h7E or Retry
						P

2087 **Table 44 GETACCMST Format 3: Incorrect Cancel**

S	7'h7E / W / ACK	GETACCMST CCC / T	Sr	Slave Addr / R / ACK	Incorrect 7-Bit Dynamic Address / PAR / T	Sr	7'h7E or Retry
						P	

5.1.9.3.17 Set Bridge Targets (SETBRGTGT)

2088 This Direct CCC (*Table 45*) is only relevant to Bridging Devices, i.e., I3C Devices that the Master knows in
2089 advance (not by inspecting BCR bits) to be Bridging Devices. An I3C Master shall only send this CCC to
2090 known Bridging Devices that accept it.

2091

Table 45 SETBRGTGT Format

S		SETBRGTGT CCC /T	Sr	Slave Addr /W/ ACK	Count /T	Describes Bridged Slave 0			Describes Bridged Slave 1 (Any additional Slaves will follow)			Sr	7'h7E
Sr	7'h7E /W / ACK					Dynamic Addr 0 /T	ID0 /T	ID0 /T	Dynamic Addr 1 /T	ID1 /T	ID1 /T		

2092 **Slave Addr** is either the Slave's original 7-bit static I²C Address, or else 7'h00 if the Slave had no such static
2093 I²C Address.

2094 **Count** is the number of Bridged "Slaves".

2095 Each described Bridged "Slave" is represented by two fields:

- 2096 · **Dynamic Address:** The 7 most significant bits (Bits [7:1]) contain the current value of the Slave's
2097 Main Master-assigned 7-bit Dynamic Address, and the least significant bit (Bit [0]) is filled with
2098 the value 1'b0. For Legacy I²C Devices, the value of the Dynamic Address shall be 7'h00.
- 2099 · **ID:** A 16-bit unambiguous identifier for the Bridged Device (two bytes)

2100 The meaning of the ID field is not defined by this Specification, and should be a contract between the Bridge
2101 and the Master (or any other entity that supplies such information to the Master). For example, the upper
2102 nibble could indicate the communication type (e.g. I²C, SPI, UART, LIN, etc.), and the lower 12 bits could
2103 be the port (as port number and Device selector).

5.1.9.3.18 Get Max Data Speed (GETMXDS)

2104 The Master uses this Direct CCC (there are two formats: *Table 46* and *Table 47*) to determine the SDR Mode
2105 data speed limitations of one Slave Device. The Master is required to use this CCC only when bit [0] of the
2106 addressed Slave Device's Bus Control Register (BCR) is set to 1'b1 (see *Table 6*). A Slave Device is required
2107 to support this CCC only when Bit [0] of its Bus Control Register (BCR) is set to 1'b1 (see *Table 6*).

2108 **Note:**

2109 *This CCC relates to bit data rates and Read Turnaround times, not data sizes. Data sizes are the*
2110 *subject of the CCCs Get Max Read Length (GETMRL, see **Section 5.1.9.3.6**) and Get Max Write*
2111 *Length (GETMWL, see **Section 5.1.9.3.5**).*

2112 The Slave Device shall return either:

- 2113 · GETMXDS Format 1: Two data bytes containing the Slave Device's Maximum Write Speed and
2114 Maximum Read Speed (see *Table 46*), or
- 2115 · GETMXDS Format 2: Five data bytes containing the Slave Device's Maximum Write Speed,
2116 Maximum Read Speed, and a three-byte Maximum Read Turnaround Time (see *Table 47*).

2117 The Slave signals which Format it is returning via the number of returned data bytes: Two bytes indicates
2118 Format 1, and five bytes indicates Format 2.

2119 The Slave's selection of Format 1 vs. Format 2 should be based upon whether Maximum Read Turnaround
2120 Time needs to be communicated to the Master Device.

2121 Interpretation of the returned fields maxWr, maxRd, and (for Format 2 only) maxRdTurn is shown in *Table*
2122 *48*, *Table 49*, and *Table 50* respectively.

21.23

Table 46 GETMXDS Format 1: Without Turnaround

S	7'h7E /W / ACK	GETMXDS CCC /T	Sr	Slave Addr /R / ACK	maxWr /T	maxRd /T	Sr	7'h7E
							P	

21.24

Table 47 GETMXDS Format 2: With Turnaround

S	7'h7E /W / ACK	GETMXDS CCC /T	Sr	Slave Addr /R / ACK	maxWr /T	maxRd /T	maxRdTurn /T	Sr	7'h7E
								P	

21.25

Table 48 maxWr Byte Format

Bits	Field	Description
7:3	Reserved	Reserved for future use by the MIPI Alliance
2:0	Maximum Sustained Data Rate for non-CCC messages sent by Master Device to Slave Device	0: f _{SCL} Max (default value) 1: 8 MHz 2: 6 MHz 3: 4 MHz 4: 2 MHz 5–7: Reserved for future use by the MIPI Alliance

21.26

Table 49 maxRd Byte Format

Bits	Field	Description
7:6	Reserved	Reserved for future use by the MIPI Alliance
5:3	Clock to Data Turnaround Time (T_{sco})	0: ≤ 8 ns (default value) 1: ≤ 9 ns 2: ≤ 10 ns 3: ≤ 11 ns 4: ≤ 12 ns 5–7: Reserved for future use by the MIPI Alliance
2:0	Maximum Sustained Data Rate for non-CCC messages sent by Slave Device to Master Device	0: f _{SCL} Max (default value) 1: 8 MHz 2: 6 MHz 3: 4 MHz 4: 2 MHz 5–7: Reserved for future use by the MIPI Alliance

21.27

Table 50 maxRdTurn Format

Byte	Field	Description
0	Least Significant Byte	Maximum Read Turnaround Time in μSeconds. 24-bit field can encode turnaround times from 0.0 seconds to 16 seconds.
1	Middle Byte	
2	Most Significant Byte	

5.1.9.3.19 Get HDR Capability (GETHDRCAP)

2128 This Direct CCC (*Table 51*) allows the Master to query one I3C Device to determine what HDR Mode(s)
2129 that Device supports. The queried I3C Device shall return a GETHDRCAP byte with the bit corresponding
2130 to each supported HDR Mode set to 1'b1, per *Table 52*.

2131 Note that the queried Device must support HDR Modes. The Master can determined by inspecting bit [5] of
2132 the Device's Bus Characteristics Register (BCR); see *Section 5.1.1.2.1* and the Command Code **Get Bus**
2133 **Characteristics Register (GETBCR)** (*Section 5.1.9.3.13*).

2134 **Table 51 GETHDRCAP Format**

S	7'h7E /W/ACK	GETHDRCAP CCC /T	Sr	Slave Addr /R/ACK	GETHDRCAP Byte /T	Sr	7'h7E
						P	

2135 **Table 52 GETHDRCAP Byte Fields**

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
HDR Mode 7	HDR Mode 6	HDR Mode 5	HDR Mode 4	HDR Mode 3	HDR Mode 2	HDR Mode 1	HDR Mode 0

2136 **Note:**

2137 *This CCC should only be sent to an I3C Device whose BCR indicates support for HDR Mode(s).*

5.1.9.3.20 Set Exchange Timing Information (SETXTIME)

2138 As detailed in *Section 5.1.8*, the SETXTIME CCC provides the framework for Master(s) and Slave(s) to
2139 exchange event timing information for purposes such as synchronizing controls, collecting or reconstructing
2140 timestamps, and specifying the timing data procedure. The SETXTIME CCC is available as both a Broadcast
2141 Command (*Table 53*) and as a Direct Command (*Table 54*).

2142 The SETXTIME CCC always includes the one-byte Defining Byte field, which acts as a sub-command
2143 identifier defining the specific function of the CCC. For some Sub-Commands (i.e., for some values of the
2144 Defining Byte field), additional Data Bytes follow. Interpretation of these additional Data Bytes depends
2145 upon the particular Defining Byte value, as detailed in *Table 55*.

2146 **Table 53 SETXTIME Format 1: Broadcast**

S	7'h7E /W/ACK	SETXTIME CCC /T	Defining Byte /T	(Optional) Data /T	Sr
					P

2147 **Table 54 SETXTIME Format 2: Direct**

S	7'h7E /W/ACK	SETXTI ME CCC /T	Sr	Slave1 Address /W/ACK	Defining Byte 1 /T	(Optional) Data 1 /T	Sr	Slave2 Address /W/ACK	Defining Byte 2 /T	(Optional) Data 2 /T	Sr
											P

21.48

Table 55 SETXTIME Defining Byte Values

Defining Byte Value	Sub-Command Function	Description	Additional Data Bytes
0x7F	Sync Tick "ST"	Marks the START condition, which is used as the reference time for the synchronization procedure. See Section 5.1.8.2 .	None
0xBF	Delay Time "DT"	Transfers the Delay Time value. See Section 5.1.8.2 .	One Byte: Delay from correct TPH Start to ST marked START
0xDF	Enter Async Mode 0	Commands all I3C Slaves and Masters on the Bus to operate in Async Basic Mode. See Section 5.1.8.3.1 . Anchor points: In-Band Interrupt ACK and T-Bit after the first In-Band Interrupt Payload byte. Upon receiving this Sub-Command, all I3C Devices shall reset all internal counts. All I3C Masters that support Timing Control shall support Async Mode 0.	None
0xEF	Enter Async Mode 1	Commands all I3C Slaves and Masters on the Bus to operate in Async Advanced Mode. See Section 5.1.8.3.2 . The Master will be sending periodic sync events on the Bus. START (not Repeated START) is the sync event. Upon receiving this Sub-Command, all I3C Devices shall reset all internal counts.	None
0xF7	Enter Async Mode 2	Commands all I3C Slaves and Masters on the Bus to operate in Async High-Precision Low-Power Mode. See Section 5.1.8.3.3 . Anchor points: The two clock edges following the event. I3C Devices need not reset any counters upon receiving this Sub-Command; instead, counters are started when an event occurs.	None
0xFB	Enter Async Mode 3	Commands all I3C Slaves and Masters on the Bus to operate in Async High-Precision Triggerable Mode. See Section 5.1.8.3.4 . Anchor points: the two clock edges following the event. I3C Devices need not reset any counters upon receiving this Sub-Command; instead, counters are started when an event occurs. Async Mode 3 is compatible with triggering events, which may be delayed by a predetermined SCL count and burst count after a trigger command. The Master keeps the SCL period uniform at least between the trigger command and the trigger event.	None
0xFD	Async Trigger (for Async Mode 3)	Used with Async Mode 3. This Sub-Command triggers active events on Slaves after a pre-determined time delay, which is set by a prior Direct CCC. High precision delay values consist of an SCL count followed by a burst count. See Section 5.1.8.3.4 .	None
0x3F	TPH	Specifies the repetition period of synchronization event. It is followed by one or more data bytes, as agreed between the Master and the respective Sensor/Slave.	One or more bytes
0x9F	TU	Specifies the Time Unit used for DT command; it is Sensor/Slave or application specific. It is followed by one byte of TU Data, as agreed between the Master and the respective Sensor/Slave.	One byte
0x8F	ODR	Specifies the Output Data Rate for the Slave.	One byte
All Other Values	(Reserved)	Available Defining Byte values for future definition of new XTIME CCC Sub-Commands	(For future definition)

5.1.9.3.21 Get Exchange Timing Support Information (GETXTIME)

2149 This Directed CCC (*Table 56*), whose use is further detailed in *Section 5.1.8*, provides the framework for the
2150 Master to query the Exchange Timing capabilities supported by the I3C Slaves. The GETXTIME CCC causes
2151 the addressed Slave to return four data bytes containing key information on supported Timing Control modes,
2152 current state, and internal oscillator/clock frequency and inaccuracy.

2153

Table 56 GETXTIME Format

S	7'h7E /W/ACK	GETXTIME CCC /T	Sr	Slave Addr /R/ACK	Supported Modes Byte /T	State Byte /T	Freq. Byte /T	Inaccuracy Byte /T	Sr	7'h7E
									P	

2154

The queried I3C Slave returns four data bytes, with the following interpretations:

2155

- **Supported Modes Byte:** Bit mask indicating which Timing Control Mode(s) the target Slave supports (see *Table 57*). If a bit is set (has value 1'b1), then that Slave supports the corresponding Timing Control Mode.

2156

2157

2158

Table 57 GETXTIME Supported Modes Byte Fields

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reserved	Reserved	Reserved	Supports Async Mode 3	Supports Async Mode 2	Supports Async Mode 1	Supports Async Mode 0	Supports Sync Mode

2159

- **State Byte:** Bit mask indicating which Timing Control Mode (if any) is currently enabled for the target Slave, and whether any counter overflows have occurred since the most recent previous check (see *Table 58*). If a Timing Control Mode bit is set (has value 1'b1), then that Slave has currently enabled the corresponding Timing Control Mode. If the Overflow bit is set (has value 1'b1), then that Slave experienced a counter overflow since the most recent previous check.

2160

2161

2162

2163

2164

Table 58 GETXTIME State Byte Fields

Bit 7	Bit 6	Bit 5	Timing Control Mode Bits				
			Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Overflow	Reserved	Reserved	Async Mode 3 Enabled	Async Mode 2 Enabled	Async Mode 1 Enabled	Async Mode 0 Enabled	Sync Mode Enabled

2165

- **Frequency Byte:** This byte represents the Slave's internal oscillator frequency in increments of 0.5 megahertz (500 KHz), up to 127.5 MHz.

2166

2167

Example: A value of 8'd20 represents an oscillator frequency of 10.0 megahertz. The value 0 is an exception, and indicates an internal oscillator frequency of approximately 32 KHz.

2168

2169

- **Inaccuracy Byte:** This byte represents the maximum variation of the Slave's internal oscillator in 1/10th percent (0.1%) increments, up to 25.5%.

2170

2171

Example: A value of 8'd25 represents a maximum frequency variation of 2.5%.

5.1.10 Error Detection and Recovery Methods for SDR

2172 The error detection and recovery methods specified in this Section are provided in order to avoid fatal
 2173 conditions when errors occur. A set of required methods is specified for I3C Slave Devices, and a separate
 2174 set of required methods is specified for I3C Master Devices. Origins for all of these SDR Error Types are
 2175 shown in *Figure 84* through *Figure 87*.

5.1.10.1 SDR Error Detection and Recovery Methods for I3C Slave Devices

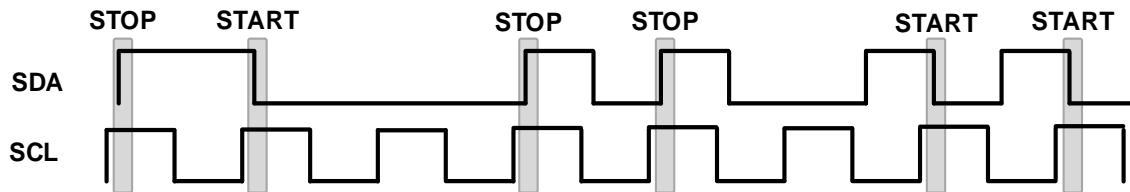
2176 The seven Error Types summarized in *Table 59* shall be supported for all I3C Slave Devices. Each Error Type
 2177 is further explained below the table.

2178 **Table 59 SDR Slave Error Types**

Error Type	Description	Error Detection Method	Error Recovery Method
S0	Broadcast Address/W (=7'h7E/W) or Dynamic Address/RW	Detect any of the following: 7'h3E / W 7'h5E / W 7'h6E / W 7'h76 / W 7'h7A / W 7'h7C / W 7'h7F / W 7'h7E / R	Enable HDR EXIT Detector and ignore all other patterns
S1	CCC Code	Parity Check, using T-Bit	Enable HDR EXIT detector and neglect other patterns.
S2	Write Data	Parity Check, using T-Bit	Enable STOP or Repeated START detector and neglect other patterns.
S3	Assigned Address during Dynamic Address Arbitration	Parity Check, using PAR Bit	Generate NACK (after PAR), then wait for another Repeated START and 7E/R to re-transmit the Provisional ID.
S4	7'h7E/R after Sr during Dynamic Address Arbitration	Detect any value other than 7'h7E/R after Sr during Dynamic Address Arbitration	Generate NACK (after 7'h7E/R), then enable STOP or Repeated START Detector and ignore all other patterns
S5	Transaction after detecting CCC	Detect illegally formatted CCC	Generate NACK (after Slave Address), then enable STOP or Repeated START Detector and ignore all other patterns
S6 (optional)	Monitoring Error	Slave detects (through monitoring) that transmitted Data differs from what it intended to transmit (Does not apply during Dynamic Address Arbitration)	Stop the transmission, then enable STOP or Repeated START Detector and ignore all other patterns

5.1.10.1.1 Error Type S0

2179 If an error occurs during Broadcast Address/W or Dynamic Address/RW, then the Slave will be unable to
2180 distinguish whether the transfer is a CCC transfer or a Private RnW transfer. For example, in the case of an
2181 ENTHDR CCC transfer the Slave is not able to know that the I3C Bus has changed to HDR Mode. A
2182 potentially fatal situation could ensue if this case is not detected and handled, because the Slave might become
2183 confused by seeing many STARTs and STOPs as illustrated in *Figure 35*, and might attempt to interpret the
2184 HDR transfer as though the I3C Bus were still in SDR Mode.



2185
2186

2187 **Figure 35 Example Waveform for Error Type S0**

2188 In order to avoid this situation, the Master shall not use any of the possible error case Addresses: 7'h7F,
2189 7'h7C, 7'h7A, 7'h76, 7'h6E, 7'h5E and 7'h3E. Except during a Dynamic Address Arbitration procedure, the
2190 Slave shall consider receipt of any of these restricted Addresses, or the receipt of 7'h7E/R, as an error and
2191 shall then ignore the rest of the signal until the HDR EXIT pattern is detected.

5.1.10.1.2 Error Type S1

2192 If the Slave detects a parity error during a CCC code, then the Slave will not be able to know that the I3C
2193 Bus has changed to HDR Mode if the CCC is ENTHDR. This is similar to the situation in Error Type S0. In
2194 order to avoid this situation, if the Slave detects a parity error during a CCC code, then the Slave shall ignore
2195 the rest of the signal, until the HDR EXIT pattern is detected.

5.1.10.1.3 Error Type S2

2196 If the Slave detects a parity error during Write Data, then the Slave shall wait for STOP or Repeated START.
2197 If the Slave detects this error after receiving a CCC, then the Slave shall either:
2198 1. Retain the CCC state until the Slave detects the end of CCC command (i.e., when the Slave is
2199 recovered by the Repeated START condition), or else
2200 2. Stop the CCC when the Slave is recovered by the STOP condition.

5.1.10.1.4 Error Type S3

2201 If the Slave detects a parity error in the PAR Bit of the Assigned Address during a Dynamic Address
2202 Arbitration procedure, then the Slave shall generate NACK (after PAR) and then wait for another Repeated
2203 START and 7E/R to re-transmit the Provisional ID.

5.1.10.1.5 Error Type S4

2204 During a Dynamic Address Arbitration procedure, if the Slave detects any value other than 7'h7E/R
 2205 following Repeated START, then the Slave shall generate NACK (after 7'h7E/R) and then wait for STOP to
 2206 exit the ENTDA mode.

2207 If the Slave detects this error after receiving a CCC, then the Slave shall either:

- 2208 1. Retain the CCC state until the Slave detects the end of CCC command (i.e., when the Slave is
 2209 recovered by the Repeated START condition), or else
- 2210 2. Stop the CCC when the Slave is recovered by the STOP condition.

5.1.10.1.6 Error Type S5

2211 If the Slave detects an illegally formatted CCC, then the Slave shall generate NACK (after the Dynamic
 2212 Address) and then wait for STOP or Repeated START. An example of an illegally formatted CCC would be
 2213 if the Slave receives the Dynamic Address/Write during the GETBCR CCC.

2214 If the Slave detects this error after receiving a CCC, then the Slave shall either:

- 2215 1. Retain the CCC state until the Slave detects the end of CCC command (i.e., when the Slave is
 2216 recovered by the Repeated START condition), or else
- 2217 2. Stop the CCC when the Slave is recovered by the STOP condition.

5.1.10.1.7 Error Type S6 (Optional)

2218 If an error occurs in a RnW Bit in a Private Read transfer, then the Write Data from the Master might conflict
 2219 with the Read Data from the Slave.

2220 The Slave should always monitor the Data it transmits. If the Slave does so, then if the monitored Data differs
 2221 from the Data the Slave intended to transmit (except for Data transferred during a Dynamic Address
 2222 Arbitration procedure), the Slave shall consider that to be an error. If the Slave detects this error, it shall stop
 2223 the transmission and then wait for STOP or Repeated START.

2224 If the Slave detects this error after receiving a CCC, then the Slave shall either:

- 2225 1. Retain the CCC state until the Slave detects the end of CCC command (i.e., when the Slave is
 2226 recovered by the Repeated START condition), or else
- 2227 2. Stop the CCC when the Slave is recovered by the STOP condition.

5.1.10.2 SDR Error Detection and Recovery Methods for I3C Master Devices

2228 The two Error Types summarized in *Table 60* shall be supported for all I3C Master Devices. Each Error Type
 2229 is further explained below the table.

2230

Table 60 SDR Master Error Types

Error Type	Description	Error Detection Method	Error Recovery Method
M0	Transaction after sending CCC	Detect illegally formatted CCC	Stop the transmission, then send STOP and retry the transmission.
M1 (optional)	Monitoring Error	Master detects (through monitoring) transmitted Data different from what it intended to transmit (Does not apply during Dynamic Address Arbitration)	Stop the transmission, then send STOP and retry the transmission.
M2	No response to Broadcast Address (7'h7E)	Master detects NACK after Broadcast Address (7'h7E) transmission	Upon detection of NACK, Master transmits HDR Exit Pattern followed by STOP

5.1.10.2.1 Error Type M0

2231 If the Master detects an illegally formatted CCC, then the Master shall stop the transmission, send STOP, and
2232 retry the transmission. An example of an illegally formatted CCC would be the Master receiving just one
2233 byte from the Slave in a GETMWL CCC code, since the Master expects two bytes.

5.1.10.2.2 Error Type M1 (Optional)

2234 **Note:**

2235 *Although this section describes methods to mitigate or handle the Type M1 error, this error is not an*
2236 *expected behavior.*

2237 If an error occurs in a RnW Bit in a Private Write transfer, then the Write Data from the Master might conflict
2238 with the Read Data from the Slave. For example, the Slave could misinterpret a Private Write transfer as a
2239 Read transfer if there is an error in the RnW Bit, and as a result Write Data from the Master would conflict
2240 with Read Data from the Slave.

2241 The Master should always monitor the Data it transmits. If the Master does so, then if the monitored Data
2242 differs from the Data the Master intended to transmit (except for Data transferred during a Dynamic Address
2243 Arbitration procedure), the Master shall consider that to be an error. If the Master detects this error, it shall
2244 stop the transmission, then send STOP and retry the transmission.

2245 If the Slave does not reply, then the Master shall transmit a pattern including HDR EXIT and STOP. This
2246 requirement applies regardless of whether the I3C Bus is in SDR Mode or in HDR Mode at the time.

2247 Example: The Master should try the following sequence of steps when attempting to recover the Slave.

2248 1. The Master transmits the Private Read several times

2249 If the Slave responds (including ACK), then the Slave temporarily did not have the Data needed
2250 for a response. If there is no response from the Slave, then proceed to the next step.

2251 2. The Master transmits a pattern including HDR EXIT and STOP

2252 If the Slave responds (including ACK), then the Slave either detected an illegal format, or detected
2253 an error. If there is no response from the Slave, then no Slave with that Address is present on the
2254 I3C Bus.

5.1.10.2.3 Error Type M2

2255 If the Master does not receive an ACK of a transmitted Broadcast Address (7'h7E), then it shall transmit the
2256 HDR Exit Pattern followed by STOP in order to recover any Slave after S0, S1, S2, S5, and S6 errors.

5.1.10.2.4 Master Error Detection and Escalation Handling

2257 If the Master does not receive an ACK of a transmitted private Message to a Slave, and if the following
2258 conditions are all true:

- 2259 · Activity State is 0 (and has been)
- 2260 · Either the Slave Device has not indicated read-turnaround delays via GETMXDS, or the Slave
2261 Device has indicated a GETMXDS period and a period longer than GETMXDS has elapsed
- 2262 · The Slave Device has not notified the Master that it will be going into a lower Activity State via a
2263 private contract In-Band Interrupt (and either GETSTATUS or a private activity state status).

2264 then the Master has the following escalation options to recover the system:

- 2265 1. If the Master is aware that the Slave might sometimes need extra time, then the Master can choose
2266 to try again after a short delay.
- 2267 2. If that fails, then the Master shall check whether the Slave is responsive by issuing GETSTATUS
2268 to the Slave. The idea here is that the Slave might be forced to NACK a private Message due to its
2269 inner system being unready, whereas GETSTATUS is a lightweight request that does not
2270 necessarily involve the inner system.
- 2271 3. If that fails, then the Master shall use M3 error handling (see *Section 5.1.10.2.4*) to emit the Exit
2272 Pattern/STOP. This ensures that the failure is not due to Slave thinking that the I3C Bus is in HDR
2273 Mode.
- 2274 4. If that fails, and if the Slave is still not responsive, then the next step depends on the value of the
2275 BCR “Offline-Capable” bit (bit [3]):
 - 2276 a. If the Slave is not Offline-Capable, then:
 - 2277 i. The Master can try slowing the SCL clock rate, or try slowing its effective rate by using
2278 duty cycle.
 - 2279 ii. If that fails, then any further escalation is outside the scope of the I3C Specification.
 - 2280 b. If the Slave is Offline-capable, then:
 - 2281 i. If the Slave is known (i.e., via a private contract) to have a long wake period, and also
2282 known to monitor the I3C Bus during that wake period, then the Master simply delays
2283 and tries later, after a delay based on the known or expected wake up time. Escalation and
2284 the GETSTATUS ensure that the Slave’s Dynamic Address was issued; that should serve
2285 as the wake trigger. Either the Slave may issue an In-Band Interrupt at a later time to
2286 notify the Master that it is ready, or else the Master may initiate the retry.
 - 2287 ii. If Slave is not known to have a long wake period and to always monitor the I3C Bus,
2288 then the Master marks the Slave as offline. The Master may then either retain the Slave’s
2289 Dynamic Address for re-use, or else discard it.

2290 The next action will be for the Slave to issue a Hot-Join request, in order to be
2291 re-joined to the I3C Bus. In the Hot-Join operation the Master will assign the
2292 Slave a Dynamic Address; the new Address may be either the same Address that
2293 the Slave used before being marked as offline, or a different Address.

5.2 High Data Rate (HDR) Modes

2294 This Section specifies the communication protocols for the three defined I3C High Data Rate (HDR) Modes.
2295 These HDR Modes are designed to transfer more data at the same Bus frequency:

- 2296 · **HDR Ternary Symbol Pure-bus (HDR-TSP) Mode**
- 2297 · **HDR Ternary Symbol Legacy-inclusive-bus (HDR-TSL) Mode**
- 2298 · **HDR Double Data Rate (HDR-DDR) Mode**

2299 **Note:**

2300 *It is important to note that the I3C Bus is always initialized and configured in SDR Mode, never in any*
2301 *of the HDR Modes. The procedure for entering an HDR Mode from SDR Mode is detailed below. The*
2302 *essential basic I3C specifications are found in **Section 5.1**, including:*

Subject	Section
Bus Configuration	5.1.1
Bus Communication	5.1.2
Bus Free Condition	5.1.3.2
Bus Idle Condition	5.1.3.4
Bus Initialization and Dynamic Address Assignment Mode	5.1.4
Hot-Join Mechanism	5.1.5
In-Band Interrupt	5.1.6
Secondary Master Functions	5.1.7
Timing Control	5.1.8
Common Command Codes (CCC)	5.1.9

2303 Details unique to the HDR Ternary Modes are specified in **Section 5.2.3**. Details common to all HDR Modes
2304 are detailed in this Section and in **Section 5.2.1**.

2305 Each HDR Mode is separate from all other HDR Modes:

- 2306 · I3C Slaves may choose to support any desired combination of HDR Modes, including no HDR
2307 Modes. If the I3C Bus enters an HDR Mode that an I3C Slave does not support, then that Slave can
2308 simply wait and watch for the common HDR Exit Pattern.
- 2309 · I3C Masters may choose to support any desired combination of HDR Modes, including no HDR
2310 Modes. However, manufacturers are generally encouraged to support all of the HDR Modes.

2311 HDR Modes have Bus-wide effect. That is, the whole I3C Bus can be put into a given HDR Mode, and once
2312 entered that HDR Mode shall remain in effect until the end of that transaction.

2313 An HDR Mode period on the I3C Bus involves five steps:

- 2314 1. The Master Broadcasts an Enter HDR Mode CCC, indicating which particular HDR Mode to
2315 enter. (See the Command Codes **Enter HDR Mode 1 through Enter HDR Mode 8 [ENTHDR1–**
2316 **ENTHDR8]** in **Section 5.1.9.3.9**).
- 2317 2. The I3C Bus switches to the requested HDR Mode.
- 2318 3. The Master issues an HDR Command, followed by HDR data sent by the Master or Slave Device.
- 2319 4. An HDR Restart Pattern or HDR Exit Pattern (see **Section 5.2.1**) is sent.
2320 · If an HDR Restart Pattern, then a new HDR Command is sent.
- 2321 5. An I3C STOP, which ends with the Bus Free Condition.

5.2.1 HDR Exit Pattern and HDR Restart Pattern

2322 Once an HDR Mode is entered, the HDR Exit Pattern is used to leave it, always exiting back to SDR Mode.
 2323 The same HDR Exit Pattern is used to exit all HDR Protocols; that Pattern does not appear in any HDR
 2324 Protocol’s normal data or command flow. All I3C Slaves shall detect and respond to the HDR Exit Pattern,
 2325 irrespective of whether the Slave supports any particular HDR Mode. The HDR Exit Pattern Detector is
 2326 specified in *Section 5.2.1.3*.

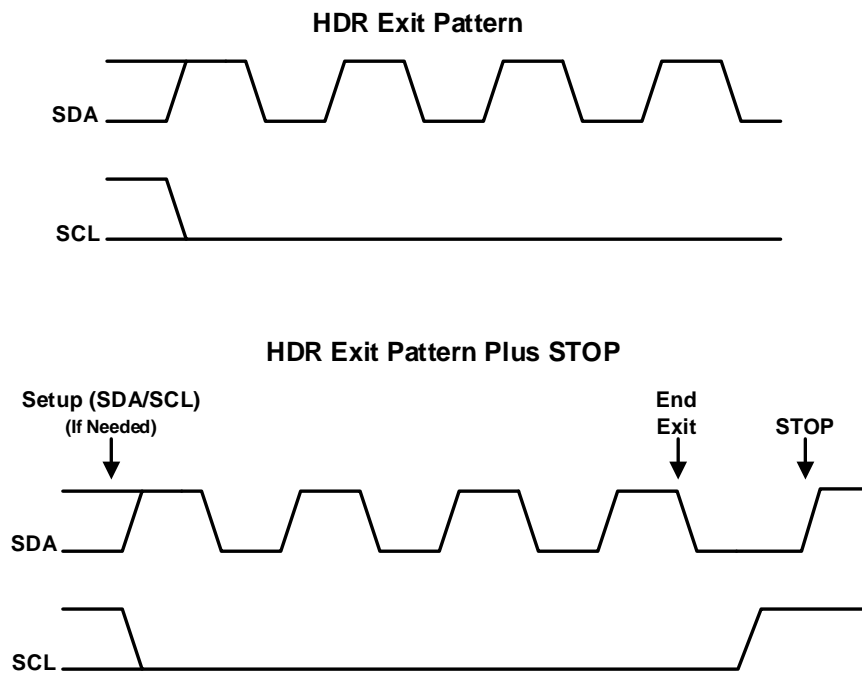
2327 As an alternative to the HDR Exit Pattern, the HDR Restart Pattern is also available. The HDR Restart Pattern
 2328 allows multiple Messages to be sent while in HDR Mode, without forcing intervening exits to SDR Mode.
 2329 That is: While the I3C Bus is in a given HDR Mode, an HDR Command can be sent to or from a Slave, and
 2330 then the HDR Restart Pattern can be used to immediately send another HDR Command to or from the Slave
 2331 (or a different Slave), without needing to exit the current HDR Mode between the HDR Commands. All I3C
 2332 Slaves shall detect and respond to the HDR Restart Pattern while operating in any HDR Mode that the Slave
 2333 supports. The HDR Restart Pattern Detector is specified in *Section 5.2.1.4*. Note that unlike the HDR Exit
 2334 Pattern, the HDR Restart Pattern is only detected by I3C Slaves that support the current HDR Mode.

5.2.1.1 HDR Exit Pattern

2335 The HDR Exit Pattern is defined thus:

- 2336 · SDA starts High, SCL starts Low
- 2337 · SDA falls (from High to Low) 4 times, while SCL remains Low (for the whole time)
- 2338 · Each SDA transition is separated by a time interval of at least t_{DIG_H} (see *Section 6.2*)
- 2339 · At the end of the HDR Exit Pattern, first SCL rises and then SDA rises. This is a normal I3C STOP.

2340 *Figure 36* illustrates an HDR Exit Pattern in the upper diagram (with edges to set SCL and SDA up into
 2341 correct state), and an HDR Exit Pattern with appended I3C STOP in the lower diagram (SCL being High
 2342 while SDA rises).



2343
2344

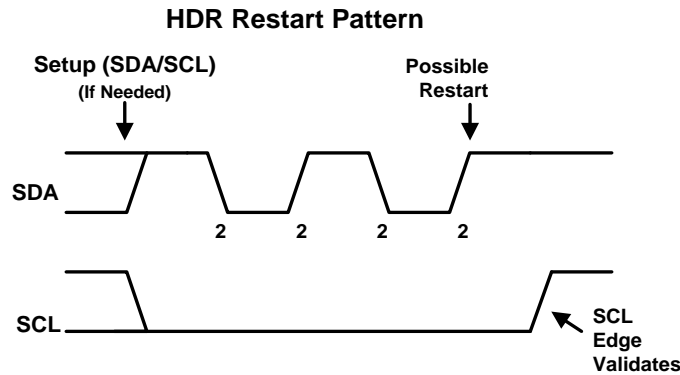
Figure 36 HDR Exit Pattern and Exit Plus STOP

5.2.1.2 HDR Restart Pattern

2345 The HDR Restart Pattern is based on a subset of the HDR Exit Pattern. It is defined thus:

- 2346 · SDA starts High, SCL starts Low (same as HDR Exit Pattern)
- 2347 · SDA toggles 4 times (fall, rise, fall, rise)
- 2348 · The next edge is SCL rising. SDA may change with SCL rising, but SCL shall rise.

2349 *Figure 37* illustrates an HDR Restart Pattern (with edges to set SCL and SDA up into correct state) along
2350 with the necessary SCL ending edge. The “2” labels represent the HDR Ternary coding, see *Section 5.2.3.1*.



2351
2352

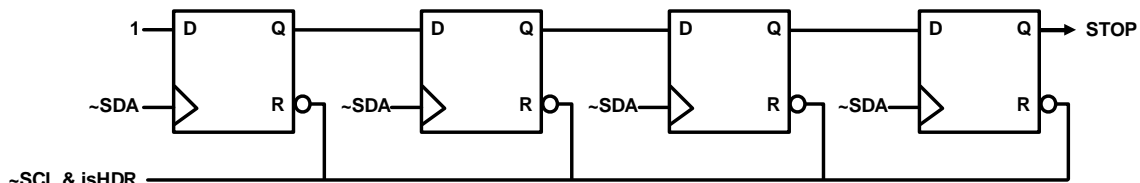
Figure 37 HDR Restart with Next Edge

5.2.1.3 HDR Exit Pattern Detector

2353 All I3C Slaves shall include an HDR Exit Pattern Detector. The HDR Exit Pattern Detector shall be enabled
2354 only after an HDR Mode is entered, and shall be disabled after an HDR Exit pattern is detected. The HDR
2355 Exit Pattern Detector may be implemented either in digital logic, or in software (bit banded).

2356 The remainder of this Section presents an example digital logic implementation in both schematic view and
2357 in RTL code.

2358 The basic logic model is that SCL is held Low (0), and so SCL High (1) shall reset the detector. It also only
2359 uses falling edges of SDA, hence treats SDA as a clock. The HDR Exit Pattern Detector schematic is shown
2360 in *Figure 38*.

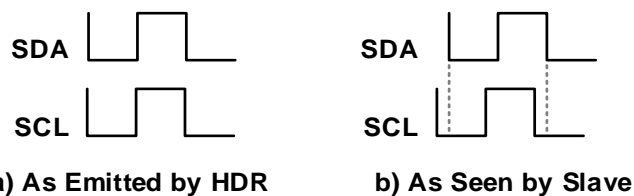


2361
2362

Figure 38 Example HDR Exit Pattern Detector (Schematic)

2363 The Detector uses an inverted version of SDA as a clock (so positive edge logic, but can use negative edge
2364 logic), and is reset when SCL is High (1), or when the block is not in HDR Mode. The asynchronous nature
2365 of the reset makes this safe, even in the presence of HDR Ternary Modes (which change SCL and SDA at the
2366 same time). This is shown in *Figure 39*. Due to the nature of SCL vs. SDA changing at the same time with
2367 Ternary Modes, the Bus Slave may see SCL change after on SDA and before the next. If the HDR Exit Pattern
2368 Detector were only using clocking logic, then it would not see any change at all (SDA posedge would always
2369 see SCL Low in this example). Because the Detector uses an asynchronous reset on SCL, a change in SCL
2370 will impact the counter, even in case (b) above. Note that SCL and SDA will still be approximately 50ns

2371 between changes. So, as shown, if SCL rises at any time, then the HDR Exit Pattern Detector shall be reset,
2372 and therefore will not mistakenly signal a false Exit.



2373

2374

Figure 39 Metastable Changes on SCL and SDA Do Not Break the Exit Pattern Detector

2375

An example RTL implementation of the above schematic is shown in *Listing 1*.

2376

Listing 1 Example RTL Code for HDR Exit Pattern Generator

```
2377 wire          scl_rst_n;
2378 wire          SDA_clk_n;          // ~SDA as clock
2379 reg[3:0]      stp_cnt;            // HDR STOP counter (shift chain)
2380
2381 assign scl_rst_n = ~iSCL & iIsInHDR; // SCL=1 resets
2382 // next uses glitch free XOR for SDA clock. Only inverts
2383 // SDA as clock if not in scan. Could add a clock mux
2384 // so uses one common clock in scan.
2385 SAFE_CLK_XOR sda_neg_clk(iSDA, ~scan_mode, SDA_clk_n);
2386
2387 // counter for 3 and 4 rising SDA when SCL is High
2388 // (else SCL resets). Whole thing held in reset if
2389 // not in HDR Mode
2390 always @ (posedge SDA_clk_n or negedge scl_rst_n)
2391   if (~scl_rst_n)
2392     stp_cnt <= 4'd0;          // SCL High or not HDR
2393   else
2394     stp_cnt <= {stp_cnt[2:0], 1'b1}; // shift chain counter
2395
2396 assign oHDR_STOP = stp_cnt[3]; // detects Exit (STOP)
```

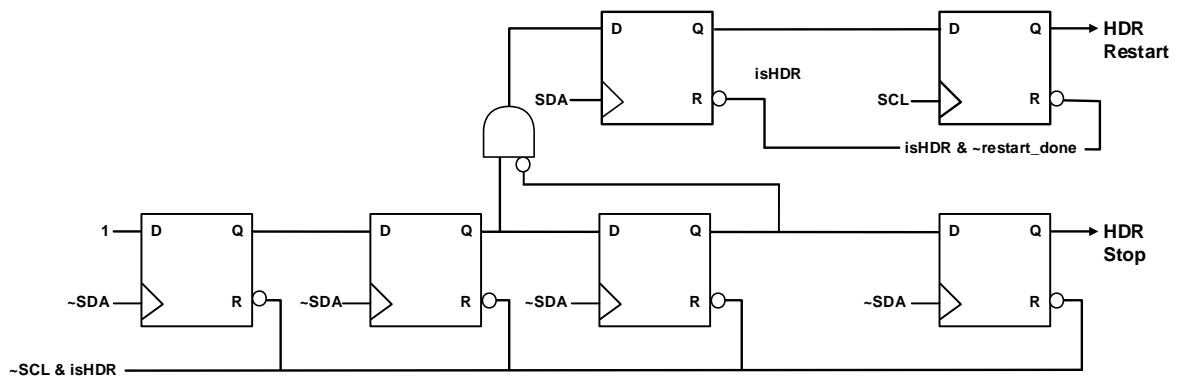

5.2.1.4 HDR Restart and Exit Pattern Detector

2397 Any I3C Slave that supports at least one HDR Mode shall include HDR Restart Pattern detection. While this
2398 function is easily incorporated into the required HDR Exit Pattern Detector, or may be part of HDR Mode
2399 support, the particular design is not mandated by this Specification (i.e. is up to the manufacturer). The HDR
2400 Restart Pattern Detector may be implemented either in digital logic or in software (bit banded).

2401 The remainder of this Section presents an example digital logic implementation in both schematic view and
2402 in RTL code, building upon the HDR Exit Pattern Detector presented in *Section 5.2.1.3*.

2403 The basic logic model is that SCL is held Low (0) and so SCL High (1) will reset the main detector. It also
2404 uses falling edges of SDA primarily, hence treats SDA as a clock. The HDR Restart is then detected only
2405 when two falling edges have been seen, and verifies the rising edge and then SCL change that is required for
2406 an HDR Restart.

2407 The combined HDR Restart and Exit Pattern Detector schematic is shown in *Figure 40*.



2408
2409

2410 **Figure 40 Combined HDR Restart and Exit Pattern Detector (Schematic)**

2411 This Detector design builds on the HDR Exit Pattern Detector. After exactly two SDA falling edges are seen,
2412 followed by a rising edge, the HDR Restart Pattern Detector checks for the rising edge of SCL. It does not
2413 matter whether SDA also falls at the same time; the rising SCL is the key. This works because even if SDA
2414 were falling (and therefore triggering the next flop in the Exit Pattern Detector), the upper-left flop will still
2415 hold 1 because it has not yet seen a rising edge on SDA.

2416 An example RTL implementation of the above schematic is shown in *Listing 2*.

2417 **Listing 2 Example RTL Code for Combined HDR Pattern Detector: Exit and Reset**

```

2418 wire          scl_rst_n;
2419 wire          restart_rst_n;
2420 wire          SDA_clk_n;          // ~SDA as clock
2421 reg[3:0]      stp_cnt;            // HDR STOP counter (shift chain)
2422 reg          poss_restart;        // possible restart
2423 reg          is_restart;
2424
2425 assign scl_rst_n      = ~iSCL & iIsInHDR; // SCL=1 resets
2426 assign restart_rst_n = ~iRestartDone & iIsInHDR;
2427
2428 // next uses glitch free XOR for SDA clock. Only inverts
2429 // SDA as clock if not in scan. Could add a clock mux
2430 // so uses one common clock in scan.
2431 SAFE_CLK_XOR sda_neg_clk(iSDA, ~scan_mode, SDA_clk_n);
2432
2433 // counter for 3 and 4 rising SDA when SCL is High
2434 // (else SCL resets). Whole thing held in reset if
2435 // not in HDR Mode
2436 always @ (posedge SDA_clk_n or negedge scl_rst_n)
2437   if (~scl_rst_n)
2438     stp_cnt <= 4'd0;          // SCL High or not HDR
2439   else
2440     stp_cnt <= {stp_cnt[2:0], 1'b1}; // shift chain counter
2441
2442 assign oHDR_STOP = stp_cnt[3]; // detects Exit (STOP)
2443
2444 // Possible Restart means exactly 2 falling edges
2445 // of SDA and then rising edge of SDA. Actual
2446 // Restart is from SCL then rising
2447 always @ (posedge iSDA or negedge restart_rst_n)
2448   if (~restart_rst_n)
2449     poss_restart <= 1'b0;    // Restart ACK or not HDR
2450   else if (stp_cnt[1] & ~stp_cnt[2]) // after 2nd fall only
2451     poss_restart <= 1'b1;    // SDA rise after 2 falls
2452   else
2453     poss_restart <= 1'b0;    // else not possible restart
2454
2455 always @ (posedge SCL or negedge restart_rst_n)
2456   if (~restart_rst_n)
2457     is_restart <= 1'b0;      // Restart ACK or not HDR
2458   else if (poss_restart)
2459     is_restart <= 1'b1;      // SCL rises after SDA does
2460   else
2461     is_restart <= 1'b0;      // else not restart

```

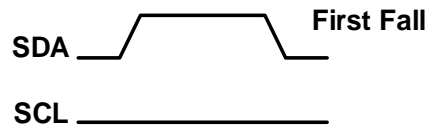
5.2.1.5 Compatibility of HDR Pattern Detection and Ternary Modes

2462 The above example circuits for HDR Restart and HDR Exit Pattern Detectors are designed to avoid certain
2463 issues that might otherwise arise in the HDR Ternary Modes. This Section discusses those issues, and how
2464 the example designs address them.

2465 HDR Ternary Modes have the special property that both SCL and SDA may be changing at the same time.
2466 As a result, I3C Slaves may see one line change slightly before the other (an interval of just a few
2467 nanoseconds). This sensitivity, typical with metastable inputs, can present problems. A successful Detector
2468 design is driven by an understanding of the allowed and disallowed triggers, including the states leading into
2469 the HDR Restart and HDR Exit.

2470 The model for the logic is:

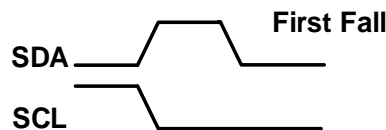
- 2471 · Once the Detector has seen exactly two SDA falling edges, the rising edge of SDA forms a
2472 possible Restart.
- 2473 · Cancels the possible Restart if a third falling edge followed by an SDA rising edge, while SCL
2474 stays Low, shall cancel that.
- 2475 · After a possible Restart, it then needs to see SCL rise to be a HDR Restart. This is because the first
2476 Symbol of an HDR ternary command will not be a Symbol 2, so SCL will change; since SCL is
2477 Low, this means it has to rise.
- 2478 · The SCL rise shall cancel the regular Exit detector since SCL High (1) resets the flops on the
2479 bottom.
- 2480 · The top two flops (Restart) only reset on exit from HDR (caused by HDR Exit/S) or HDR logic
2481 acknowledging the HDR Restart (restart_done).
- 2482 · To see why the Detector could not be misaligned, consider that leading into this the Bus has to
2483 have SCL Low (so flops not reset) and SDA High (so detector reacts to first falling edge of SDA).
2484 So, the preceding states could be:
 - 2485 · SCL was already Low; SDA moves Low to High, whether from last Symbol or forced edge (to
2486 put into right state), see *Figure 41*. This is rising edge of SDA, so OK.



2487
2488

Figure 41 SDA Moves Low to High

- 2489 · SCL goes Low while SDA goes High, whether from last Symbol of forced edge (to put into
2490 correct state), see *Figure 42*.

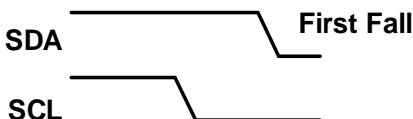


2491
2492

Figure 42 SDA and SCL Changing Metastable

- 2493 · Simultaneous change could be seen as SCL falling edge and then SDA rising edge (so
2494 SCL reset releases, then rising edge – since rising edge, it is ignored).
- 2495 · Simultaneous change could be seen as SDA rising edge and then SCL falling edge (so
2496 SCL reset releases after SDA is High, so no effect).

- 2497 · SCL goes Low while SDA was already High, whether from last Symbol or forced edge (to put
2498 into right state), see **Figure 43**. This would release the reset and have no other effect.
2499



2500

2501

Figure 43 SCL Goes Low While SDA Already High

- 2502 · For the Cancel case, the possible-Restart flop only goes back to Q=0 if it sees another falling SDA
2503 edge (third) and then rising after. So, even if the first edge after the Restart is SCL rising edge
2504 while SDA is falling edge, it does not matter what order seen (metastable input), since it would
2505 need to see SDA fall and then rise in order to cancel.

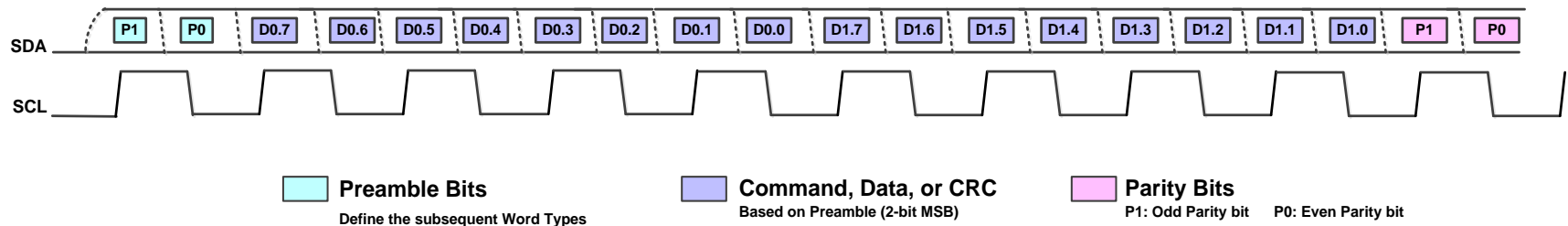
5.2.2 HDR Double Data Rate Mode (HDR-DDR)

2506 Like SDR Mode, HDR-DDR Mode uses SCL as a clock; however unlike SDR, Data and Commands change
2507 SDA on both SCL edges (when High and when Low), effectively doubling the data rate. By contrast, in SDR
2508 Mode SDA is changed only when SCL is Low. Since SDR Mode defines it as a START or a STOP for SDA
2509 to change while SCL remains High, HDR-DDR Mode is classified as an HDR Mode in order to prevent
2510 confusion.

2511 HDR-DDR moves data by Words. A Word generally contains 16 payload bits, as two bytes in a byte stream,
2512 and two parity bits. Four HDR-DDR Word Types are defined: Command Word, User Data, CRC Word, and
2513 Reserved Word.

2514 The HDR-DDR Protocol precedes each 18-bit Word with a 2-bit Preamble, for a total of 20 bits per Word.
2515 The Preamble:

- 2516 · Indicates the type of data that follows: either Command, Data, or CRC
2517 · Allows the Master to terminate a Read, and to determine whether the Slave is willing to respond to
2518 a Read. This is controlled such that the first bit is Parked High and the other side can choose either
2519 to drive Low, or to leave High.



2520
2521

Figure 44 HDR-DDR Format

2522 Preamble bit interpretation depends on the context, as shown in **Table 61**. The context controls the roles of
 2523 Master and Slave, such that for Data, the Slave or Master drives the second Preamble bit after the other I3C
 2524 Device has Parked High. Note that a Preamble value of b00 is not used and is reserved for future use.

2525

Table 61 HDR-DDR Preamble Values

Context	Preamble Value and Interpretation			
	2'b00	2'b01	2'b10	2'b11
After EnterHDR	Reserved for Future Use	Command Word follows	-	-
After Read CMD		-	Slave ACK, Data follows	Slave NACK, Aborted
After Read DATA		CRC Word follows	Master Aborts, Slave yields. <i>Master drives second 0.</i>	Data follows. <i>Master does not drive second bit.</i>
After Write CMD		-	Data follows	-
After Write DATA		CRC Word follows	-	Data follows

2526 HDR-DDR Mode defines four Word Types:

- 2527 • **Command Word.** The Preamble before a Command Word shall have the value 2'b01. The length
 2528 of a Command Word shall be 18 bits (16 value bits and 2 parity bits). HDR-DDR Mode Command
 2529 Codes allow up to 128 Write or Read Commands, respectively (see **Section 5.2.2.2**).
- 2530 • **Data Word.** The Preamble before a Data Word shall contain the value 2'b01, 2'b10, or 2'b11 (see
 2531 **Table 61** and **Figure 45**). The length of a normal Data Word shall be 18 bits (16 value bits as two
 2532 bytes, and 2 parity bits).
- 2533 • **CRC Word.** The Preamble before a CRC Word shall have the value 2'b01. The length of a CRC
 2534 Word shall be 11 clocked bits (4 bit 4'hC token, 5 bits of CRC5 value, 1 bit setup to prepare the
 2535 Bus for the following HDR Exit or HDR Reset, and 1 clock for High-Z by the Slave). The value of
 2536 the upper nibble (first 4 bits after the Preamble) of a CRC Word shall be 4'hC. The CRC5
 2537 algorithm is specified in **Section 5.2.2.5**. There is no parity checking for a CRC Word. Following
 2538 transmission of a CRC Word, an I3C Device shall set up SCL and SDA to allow for an HDR
 2539 Restart Pattern or HDR Exit Pattern.
- 2540 • **Reserved Word.** The Preamble before a Reserved Word shall have the value 2'b01. The length of
 2541 a Reserved Word shall be 18 bits (16 value bits and 2 parity bits). The value of the upper nibble
 2542 (first 4 bits after the Preamble) of a Reserved Word shall be 4'hD, 4'hE, or 4'hF. I3C Devices shall
 2543 treat reception of a Reserved Word as an error, and not use any received Reserved Word.

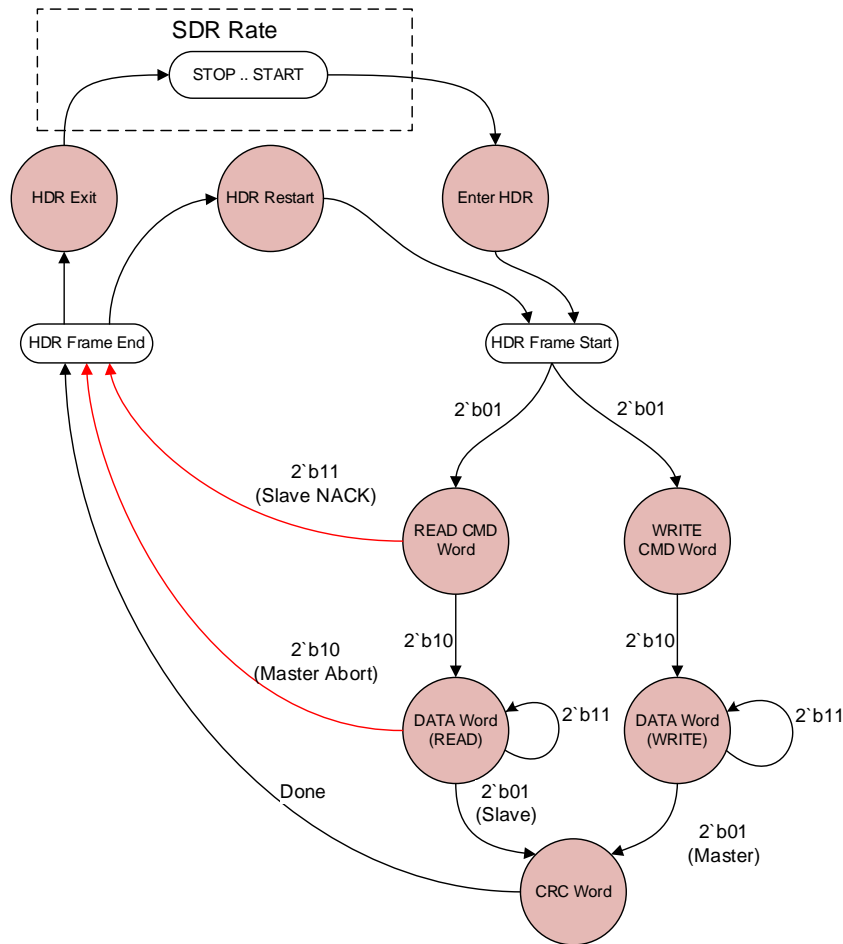


Figure 45 DDR Preamble Bits State Diagram

5.2.2.1 HDR-DDR Overview

The HDR-DDR Protocol uses the same signaling as SDR, but operates with SDA changing after each SCL edge.

- A normal HDR-DDR Read Request shall be composed of a Command Word from the Master, followed by one or more Data Words from the Slave, followed by one CRC Word from the Slave as shown in *Figure 47*.
- An HDR-DDR Write Request shall be composed of a Command Word, followed by zero or more Data Words, and one CRC, all from the Master, as shown in *Figure 46*.
- A NACKed HDR-DDR Read Request shall be composed of a Command Word, followed by the Slave not ACKing (driving Low SDA) in the second pre-amble bit, before what would be the first Data Word.

Each Command Word and each Data Word is 20 bits in length (20 clock edges) including a two-bit Preamble. At 10MHz the raw bit rate is 20 Mbps; at 12.5 MHz the raw bit rate is 25 Mbps. Because each Word includes two Preamble bits and two Parity bits, the real data rate (considering only the two bytes in the 16 payload bits per Word) is the raw rate times 16/20, or 16 Mbps at 10 MHz (20 Mbps at 12.5 MHz), which is a measure of the 16-bit data rate.

HDR-DDR Command Words indicate the direction of data movement: either Write (Master to Slave), or Read (Slave to Master). After the Command Word, zero or more Data Words are sent by the Master or by the Slave (unless NACKed) until done, followed by the CRC Word (unless NACKed). Finally, the Master issues

2564 either an HDR Restart Pattern or an HDR Exit Pattern. The Master may also terminate a Read prematurely,
2565 per **Section 5.2.2.3.2**, however this is not a normal end for a Read, and should be used sparingly.

2566 In HDR-DDR Mode, just as in SDR Mode, only the I3C Bus Master drives the SCL line. For Commands,
2567 the SDA line is driven by the Master. For Data, the SDA line is driven either by the Slave or by the Master,
2568 depending on the Command direction. Bus Turnaround behavior is specified in **Section 5.2.2.3**.

2569 The common format used by HDR-DDR Command Words, Data Words, and Reserved Words is illustrated
2570 in **Table 62**. The Command details are explained in **Section 5.2.2.2**. The format used by HDR-DDR CRC
2571 Words is a subset of the Command Word and the Data Word (see **Section 5.2.2.5**).

2572

Table 62 HDR-DDR Word Format: Command, Data, Reserved

Preamble	Payload	Parity (not the CRC)	
2 bits	16 bits	1 bit	1 bit
2'b00: Not used 2'b01: Command or CRC 2'b10, 2'b11: Data or Abort	Command Code or Data Values	XOR of Odd index Payload bits	XOR of 1 and Even index Payload bits

2573 The two parity bits are formed from the payload bits, using an XOR function:

2574 · P1 = Parity of the odd index Payload bits:

2575
$$D [15] \wedge D [13] \wedge D [11] \wedge D [9] \wedge D [7] \wedge D [5] \wedge D [3] \wedge D [1]$$

2576 · P0 = Parity of the even index Payload bits and 1:

2577
$$D [14] \wedge D [12] \wedge D [10] \wedge D [8] \wedge D [6] \wedge D [4] \wedge D [2] \wedge D [0] \wedge 1$$

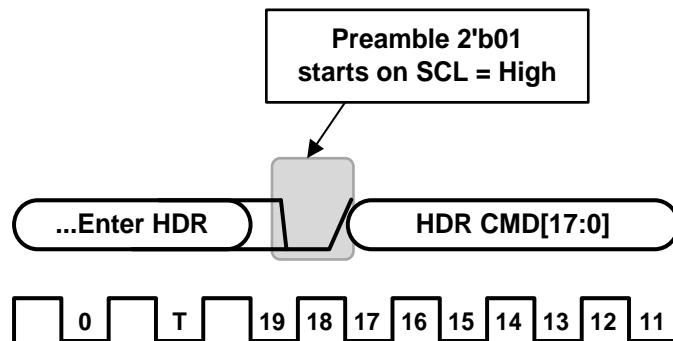
2578 **Table 63** summarizes the format of all four HDR-DDR Word Types.

2579

Table 63 HDR-DDR Word Formats

Word Type	Preamble	Payload		Parity		Notes
	2 bits	16 bits		1b	1b	
Command	2'b01	15:8	Command Code (8 bits)	P1	P0	Command may follow only EnterHDR or HDR Restart. Command Codes: Write: 8'h00 to 8'h7F Read: 8'h80 to 8'hFF
		7:1	Slave Address (7 bits)			
		0	Reserved (1 bit)			
Data	2'b10 2'b11	15:8	First Data Byte (8 bits)	P1	P0	0 or more Data words may follow Command. Only CRC may follow Data.
		7:0	Second Data Byte (8 bits)			
CRC	2'b01	15:12	4'hC (token value) (4 bits)	Not Used		CRC value ends at bit 6, but bit 5 allows High-Z, then HDR Restart or HDR Exit begins with no clocks. Parity is not used. Setup bit puts SDA High in preparation for HDR Exit/Restart
		11:7	CRC5 value (5 bits)			
		6:5	(2 bits) Setup bits to prepare for HDR Restart or HDR Exit. Slave parks SDA High.			
		4:0	Reserved (5 bits), No clocks			
Reserved	2'b01	15:12	4'hD to 4'hF (4 bits)	P1	P0	Reserved for future use
		11:0	Reserved (12 bits)			
–	2'b00	Do not use Preamble value 2'b00				

2580 HDR-DDR Mode is entered in the standard way, using the Enter HDR CCC (see **Section 5.1.9.3.8**). After the
2581 Enter HDR CCC and its ninth bit (the T-Bit), the SCL falling edge begins the HDR Mode. The first HDR-
2582 DDR bit starts on the next SCL rising edge; that is, the Master drives SDA when SCL is Low, and the Slave
2583 samples SDA on the SCL rising edge. This allows the entry to HDR-DDR Mode, as shown in **Figure 46**.



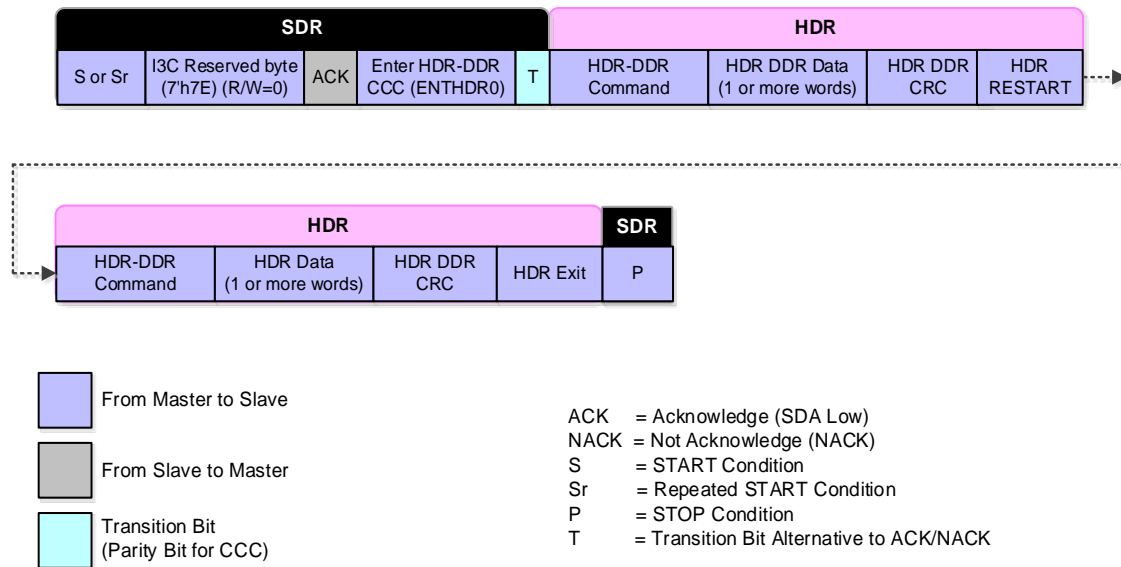
2584

2585

Figure 46 Unused Bit Before Preamble of First HDR-DDR Command

2586 Once in HDR-DDR Mode the Master issues a Command Word and then zero or more Data Words. In HDR-
2587 DDR Mode, Commands shall be issued only by a Master. Data Words may be issued by a Master or by a
2588 Slave, depending on the particular Command (Write or Read).

- 2589 **Figure 47** illustrates a typical HDR-DDR Mode Frame with two HDR Commands and their associated data:
 2590 · I3C START
 2591 · I3C CCC to Enter HDR-DDR Mode
 2592 · After entering HDR-DDR Mode, there is one HDR-DDR Command Word followed by one HDR-
 2593 DDR Data Word, and then an HDR-DDR CRC Word
 2594 · Then an HDR Restart Pattern
 2595 · then another HDR-DDR Command Word, HDR-DDR Data Word, and HDR-DDR CRC Word
 2596 · and finally HDR Mode is ended via the HDR Exit Pattern
 2597 · followed by I3C STOP



2598
2599

Figure 47 Typical HDR-DDR Mode Frame

5.2.2.2 HDR-DDR Command Coding

2600 In HDR-DDR Mode a Command Word follows the initial Enter HDR CCC (and any HDR Restart). This
2601 Command Word uses the normal 18-bit model. **Table 64** illustrates the Command Word format for HDR-
2602 DDR Mode.

2603

Table 64 HDR-DDR Command Word Format

Bits	Field	Size (bits)	Notes
15	Read/Write	1	1=Read (Slave to Master) 0=Write (Master to Slave)
14:8	Command Code	7	128 possible Write commands 128 possible Read commands
7:1	Slave Address	7	Same Dynamic Address as used in I3C SDR Protocol
0	Write Reserved Read Parity Adjustment	1	For Read, ensures that P0 contains 1, which allows for easier handoff ¹
Note: 1) Because P0 is the XOR of the even index Payload bits and 1, it is equal to: $\text{XOR_outer}(1, \text{this bit}, \text{XOR_inner}(2, 4, 6, 8, 10, 12, 14))$ As a result, this bit should be set to the result of XOR_inner().			

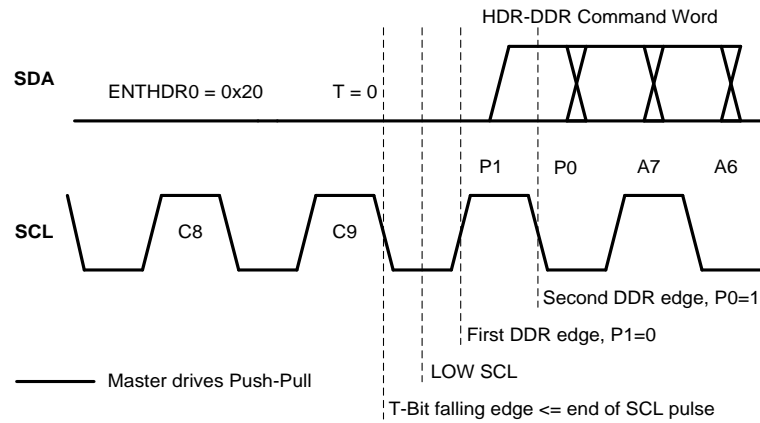
2604 Regarding bits 15:8 together, the possible Command Code spaces for Read Commands and Write Commands
2605 in HDR-DDR Mode are illustrated in **Table 65**.

2606

Table 65 Read and Write Command Spaces for HDR-DDR Mode

Command Codes	Command Functions	
0x00 – 0x7F	Write Commands (128 possible)	0x00 – 0x1F: Reserved for I3C Definition – 32 Commands
		0x20 – 0x7F: Reserved for Vendor Definition – 96 Commands
0x80 – 0xFF	Read Commands (128 possible)	0x80 – 0x9F: Reserved for I3C Definition – 32 Commands
		0xA0 – 0xFF: Reserved for Vendor Definition – 96 Commands

2607 Signal diagrams for starting the HDR-DDR Command Codes are shown in *Figure 48* and *Figure 49*. *Figure*
2608 *48* shows the HDR-DDR Command Code after ENTHDR0, and *Figure 49* shows the HDR-DDR Command
2609 Code after HDR Repeated Start Pattern.



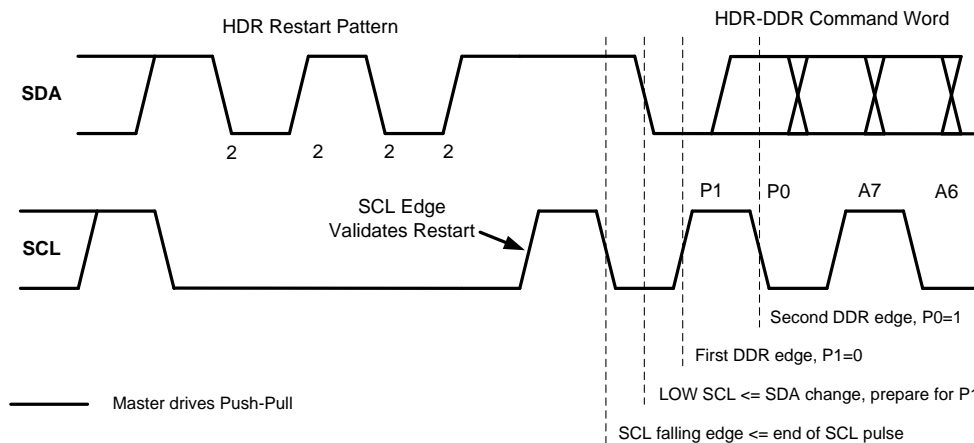
2610

2611

Figure 48 HDR-DDR Command Code After ENTHDR0

2612 As *Figure 48* shows:

- 2613 1. First the SCL pulse is ended by its falling edge (as is necessary for the Slaves' logic design).
- 2614 2. Then on the SCL Low that follows, the Master positions the SDA (i.e. sets SDA either High or
- 2615 Low) as necessary to be read on the first SCL rising edge, using Push-Pull timing.



2616

2617

Figure 49 HDR-DDR Command Code After HDR Repeated Start Pattern

2618 As *Figure 49* shows:

- 2619 1. After the SCL rising edge for validating the HDR Repeated Start pattern, an SCL falling edge is
- 2620 provided. The result is an SCL pulse, identical to the SCL pulse of T-Bit (see *Figure 48*).
- 2621 2. Any possible changes on SDA are ignored until the SCL rising edge that follows (labeled 'First
- 2622 DDR edge').
- 2623 3. On the SCL Low following the SCL pulse falling edge, the Master positions the SDA (i.e. sets
- 2624 SDA either High or Low) as necessary to be read on the first SCL rising edge, using Push-Pull
- 2625 timing.
- 2626 4. The waveform that follows is identical to the same segment in *Figure 48*.

5.2.2.3 HDR-DDR Bus Turnaround

- 2627 There are three points of Bus Turnaround (between Master and Slave) with HDR-DDR Mode:
- 2628 · After a Read Command from the Master, the Slave shall drive SDA to indicate that the Slave plans
 - 2629 to return Data (see *Section 5.2.2.3.1*).
 - 2630 · After the end of Data from the Slave for a Read Command, the Master shall drive SDA again (see
 - 2631 *Section 5.2.2.3.2*).
 - 2632 · There is an optional Bus Turnaround before Data Words are returned by the Slave, allowing the
 - 2633 Master to prematurely terminate a Read (see *Section 5.2.2.3.3*).

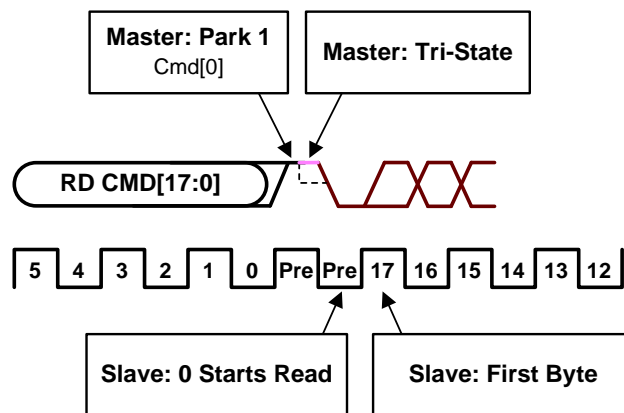
5.2.2.3.1 Command to Read Data from Slave

2634 In response to a Read Command in HDR-DDR Mode, a Slave shall either return one or more Data Words, or
2635 else ignore the Read Command. If the Command asks the Slave to return one or more Data Words, then the
2636 I3C Bus Turnaround occurs on the Preamble value 2'b10 immediately preceding the Data, as shown in

2637 *Figure 50*.

2638 To avoid problems, HDR-DDR Mode allows the Master to detect that the Slave is not present (or is not
2639 responding). To do this, the Master Parks the SDA line High on the last bit of the Read Command; this can
2640 be assured by using bit 0 of the Command Word (the Parity Adjustment bit) to ensure that parity bit P0 (last
2641 bit of the Word) has the value 1 (High). The Master then High-Zs the SDA line in the first bit of the Preamble
2642 (when SCL is Low), which stays High due to the weak Pull-Up resistor. If the SDA line is not Low by the
2643 end of the SCL High period (the second bit of the Preamble), then the Master shall assume that the Slave is
2644 not responding, and drive an HDR Restart Pattern or HDR Exit Pattern.

2645 Otherwise, the Slave drives the SDA line Low (0) as the required second bit of the Preamble, indicating Data,
2646 and then the Slave transmits the Data Words through to the CRC. The procedure for ending the data stream
2647 is specified in *Section 5.2.2.3.2*.



2648
2649

Figure 50 Start of HDR-DDR

5.2.2.3.2 End of a Read Command Message

2650 The number of Data Words associated with a Command is in many cases a contract between the Master and
 2651 Slave; that is, it depends on the particular Command Code. However, the HDR-DDR Protocol supports the
 2652 transmission of variable length Data by the Master (for a Write) by the Slave (for a Read). Additionally, a
 2653 Master may terminate a Read (see *Section 5.2.2.3.3*).

2654 For a Write (Master to Slave), the Slave shall know that Data transmission is done when a CRC Word
 2655 followed by the HDR Restart Pattern or HDR Exit Pattern is detected.

2656 For a Read (Slave to Master), the Slave shall issue a CRC Word upon conclusion of the data transmission,
 2657 and then shall Park the SDA line High on the first bit after the CRC 5-bit value, followed by tri-stating the
 2658 SDA line. The Master shall see the CRC Word, and then expect to see the 1 (Parked). The Master shall then
 2659 make sure that the SCL line is Low, and then transmit an HDR Restart Pattern or HDR Exit Pattern, after the
 2660 Slave has High-Z'ed the SDA line.

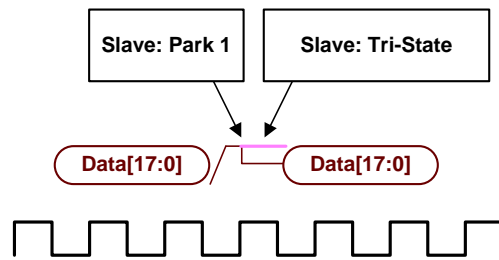
5.2.2.3.3 Master Termination of a Read Command Message

2661 For HDR-DDR Mode the normal model is that the Slave ends a Read with a CRC Word, and then hands I3C
 2662 Bus control back to the Master, as per *Section 5.2.2.3.2*.

2663 However HDR-DDR Mode also allows the Master to terminate a Read prematurely if necessary (for example,
 2664 if there is an unexpected need to regain the Bus). Premature termination is accomplished using the two
 2665 Preamble bits that the Slave transmits before each Data Word. As per *Table 63*, the Preamble transmitted
 2666 between Data Words starts with the SDA line High while the SCL line is also High. Then the Slave sets the
 2667 SDA line High-Z when the SCL line goes Low.

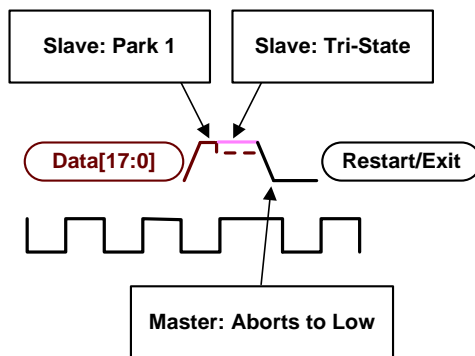
2668 To allow the Data Word transmission to continue normally, the Master simply does nothing, as shown in
 2669 *Figure 51*.

2670 To instead terminate the Data Word, the Master holds the SCL High period and drives the SDA line Low, as
 2671 shown in *Figure 52*. When the Slave sees that the SDA line is Low, it shall yield SDA control and consider
 2672 the Read operation terminated. The Master shall then issue an HDR Restart Pattern or HDR Exit Pattern.



2673
2674

Figure 51 Master Allows Normal Read



2675
2676

Figure 52 Master Terminates Read

5.2.2.4 HDR-DDR Error Detection

2677 Four error types are defined for HDR-DDR Mode:

- 2678 1. Framing – the Preamble 2-bits before Command and Data shall be valid values per *Table 65*. This
2679 supports a positional error detection mechanism:
- 2680 · A Command Word shall always follow the Enter HDR CCC and HDR Restart Pattern, and shall
2681 never appear in any other position. It is an error condition for a Command Word to appear in
2682 any other position, or to be missing where expected.
 - 2683 · A Data Word shall always follow either a Command Word or another Data Word, and shall
2684 never appear in any other position. It is an error condition for a Data Word to appear in any
2685 other position, or to be missing where expected.
 - 2686 · A single CRC Word shall always follow the last Data Word for a Command, such that it ends
2687 the Message. As a result, a CRC Word shall always be followed by either the HDR Restart
2688 Pattern or the HDR Exit Pattern. It is error condition for a CRC Word to appear in any other
2689 position, or to be missing where expected.
 - 2690 · The first nibble of a valid CRC shall contain the allowed value 4'hC. Any other value in the
2691 first nibble should be considered a framing error.
- 2692 2. Parity encoding (for transmitters) parity checking (for receivers) shall be performed on all
2693 Command Words and Data Words. Mismatched parity is an error condition.
- 2694 3. CRC5 encoding (for transmitters) and CRC5 checking (for receivers) shall be performed on all
2695 payload bits for Command Words and Data Words. Mismatched CRC is an Error.
- 2696 4. NACK by the Slave on a Read command is not a normal behavior (ACK is normal). The Master
2697 may choose to treat a NACK on a Read command as a possible framing error, and take the actions
2698 detailed below.

2699 Any error detected by the Slave should result solely in waiting for HDR Exit or HDR Restart to be safe.

2700 If the Master detects an error in a Read, then it shall issue SCL clocks until 19 SCL clocks (38 bits) have
2701 been seen with SDA High. Because a continuous High run of this length cannot occur in a valid data
2702 transmission, this ensures that the Slave has released the line. The Master shall then Park SCL Low, and emit
2703 either an HDR Exit Pattern or an HDR Restart pattern using SDA.

2704 **Note:**

2705 *The Master can choose to treat the NACK of a Read command as a possible line error, and therefore*
2706 *use the same approach in order to ensure that the Slave is not driving the Bus.*

2707 If an error occurs in the RnW Bit during a Private Read Transfer, then the Write Data from the Master might
2708 conflict with the Read Data from the Slave. Both the Master and the Slave should always monitor the Data
2709 that they each transmit. If the Master or Slave performs such monitoring, and if the monitored Data differs
2710 from the Data that the Master or Slave intended to send (except for Data transferred during a Dynamic
2711 Address Arbitration procedure), then this shall be considered an error. After detecting this error, both the
2712 Master and the Slave can stop the transmission and then the Slave wait for HDR EXIT or HDR Restart. After
2713 the Master sends HDR EXIT, it can retry the transmission.

5.2.2.5 HDR-DDR CRC5 Algorithm

2714 The CRC5 value is computed on a complete message, including the Command Word and all Data Words.

2715 · For a Command Word, the CRC5 is computed based on the value of the 16-bit payload, including:

2716 · The Read vs. Write bit

2717 · The Command Value

2718 · The Slave Address, and

2719 · The lowest-order bit (Write Reserved and Read Parity adjustment).

2720 · For Data Words, the CRC5 is computed based on all Data 16 bit values transmitted for that

2721 Command.

2722 The CRC5 value is initialized to 0x1F. The CRC5 polynomial is the same as used in *[USB01]*:

$$2723 \quad \text{CRC5} = X^5 + X^2 + X^0$$

2724 The reason why CRC5 is adequate is that most errors will be caught by the Parity bits, or by incorrect
 2725 Preamble bits. The Parity bits will detect runs of one, two, three, or more errors in a row. Any Clock errors
 2726 not detected by the Parity bits would generally be detected as framing errors. That is, the Preamble would
 2727 not match the allowed patterns, including 2'b01 leading into the CRC. As a result, any error that could be
 2728 missed by the Parity bits will result in a shift of the CRC polynomial, and therefore will be detected as a CRC
 2729 error. There would have to be two separated errors in the same Word, with either both errors occurring in
 2730 even-index bits or with both errors occurring in odd-index bits, to produce correct Parity bits. However these
 2731 two errors would not correct the CRC, and as a result would still be detected. Even with two such error pairs
 2732 in two different Words, the probability of producing the same CRC value is very low, even though the CRC5
 2733 has only 32 possible values. Since any higher error density would render the I3C Bus generally unusable, it
 2734 can be concluded that the protection that CRC5 provides is sufficient for the use cases addressed by this
 2735 Specification.

2736 CRC5 is specified in *[USB01]*. Two versions of the logic for computing the CRC5 are shown below:

2737 · First, for one bit at a time,

2738 · Second, for the more practical situation of two bits at a time (i.e., on SCL falling edge).

2739

2740 // CRC5 builds from each bit as it comes in, using the registered SDA_r.

2741 // The next_CRC5 is registered on each cycle as CRC5[4:0].

2742 // But note that it would have to be on each SCL edge, so two

2743 // alternating buffers would be needed.

2744 assign next_crc0 = SDA_r ^ CRC5[4];

2745 assign next_CRC5[4:0] = {CRC5[3:2], next_crc0^CRC5[1],

2746 CRC5[0], next_crc0};

2747

2748 // CRC5 builds from 2 bits at a time (registered d_rise_r and live d_fall),

2749 // on falling edge of SCL (by convention - could be on rising instead).

2750 // The d_fall could be d_fall_r, but then order must match input order

2751 // of the data bits.

2752 // The CRC therefore processes the two bits by combining the 2 bit shifts.

2753 // The next_CRC5 is registered on each cycle as CRC5[4:0]

2754 assign next_CRC5 = {CRC5[2], d_rise_r^CRC5[4]^CRC5[1], d_fall^CRC5[3]^CRC5[0],

2755 d_rise_r^CRC5[4], d_fall^CRC5[3]};

5.2.3 HDR Ternary Modes (HDR-TSP and HDR-TSL)

2756 I3C defines two HDR Modes that use Ternary Coding:

- 2757 · HDR-TSP: Ternary Symbol for Pure Bus (no I²C Devices)
- 2758 · HDR-TSL: Ternary Symbol Legacy-inclusive-Bus

2759 These HDR Ternary Modes are entered in the standard way (see *Section 5.1.9.3.9*), followed by a Command
2760 and then zero or more Data Words. In HDR Ternary Modes, Commands shall be issued only by a Master.
2761 Data Words may be issued by a Master or by a Slave, depending on the particular Command (Write or Read).

2762 *Figure 53* illustrates a typical HDR Ternary Mode Frame with two HDR Commands and their associated
2763 data:

- 2764 · I3C START
- 2765 · I3C CCC to Enter HDR Ternary Mode X
- 2766 · After entering HDR Ternary Mode, there is one HDR Command followed by HDR Data
- 2767 · Then an HDR Restart Pattern
- 2768 · then another HDR Command and HDR Data
- 2769 · and finally HDR Mode is ended via the HDR Exit Pattern
- 2770 · followed by I3C STOP

I3C			Msg1			Msg2			I3C
START	Brdcst	EnterHDRx	HDR Cmd	HDR Data	HDR Restart Pattern	HDR Cmd	HDR Data	HDR Exit Pattern	STOP

2771 **Figure 53 Typical HDR Ternary Mode Frame**

5.2.3.1 HDR Ternary Signaling and Coding Protocol

2772 The HDR Modes (HDR-TSP and HDR-TSL) use different signaling and coding protocol models from the
2773 rest of I3C, relying on a three-way (or ‘Ternary’) state transition model to achieve higher information density
2774 on the two-wire Bus.

5.2.3.1.1 Ternary Signaling

2775 HDR Modes HDR-TSP and HDR-TSL use Ternary signaling on the two wires. Ternary signaling effectively
2776 communicates 1.5 data bits per time event, which is 50% more than non-Ternary I3C signaling.

2777 This higher effective data rate is achieved by using both lines (SCL and SDA) for data. By contrast, I²C and
2778 non-Ternary Modes in I3C use the SCL line only as a clock and the SDA line only as data. In Ternary signaling
2779 the clock signal is implicit whenever either line (or both lines at once) changes level.

2780 The name ‘Ternary’ comes from the fact that at each line state change, exactly three transitions are possible.
2781 The transitions are then used as coefficients for base-3 (ternary) numbers.

2782 The three possible line transitions are:

- 2783 1. Only the SCL line changes
- 2784 2. Only the SDA line changes
- 2785 3. Both SCL and SDA change

2786 Each such transition is captured as a Symbol with three possible values: 0, 1, or 2. We use the term ‘Symbol’,
2787 rather than ‘data’, because the encoded data that a given Symbol represents will vary depending upon the
2788 previous Symbols.

2789 The two defined I3C HDR Ternary Modes (HDR-TSP and HDR-TSL) allow different Slave types to appear
2790 on the I3C Bus:

- 2791 · HDR-TSP (Ternary Symbol for Pure Bus) is used when only I3C compliant Slaves are present on
- 2792 the I3C Bus. As a result, all Slaves on the Bus are guaranteed have an HDR Exit Pattern Detector
- 2793 (though not all Slaves will necessarily support HDR-TSP Mode).
- 2794 · HDR-TSL (Ternary Symbol Legacy inclusive Bus) is used (and is usable) when at least one
- 2795 qualified Legacy I²C Slaves is also present (along with I3C Slaves) on the I3C Bus.

2796 Qualified Legacy I²C Slaves for HDR-TSL Mode shall be limited to those that both A) have a true 50ns Spike
 2797 Filter, and B) properly ignore SCL being High for less than 50ns as detailed in **Section 5.1.1**. An I3C Bus
 2798 with Legacy I²C Slaves not meeting these criteria will not successfully function in HDR-TSL Mode.

2799 HDR-TSP Mode and HDR-TSL Mode work similarly, with two small differences:

- 2800 · HDR-TSL shall operate faster than 10MHz. Normal speeds are 12MHz or 12.5MHz, so that edges
- 2801 are approximately 40ns apart.
- 2802 · During HDR-TSL SCL it is not allowed to stay on High position longer than ~40ns. As a result a
- 2803 Legacy I²C Slave with a true 50ns filter sees SCL only as Low.

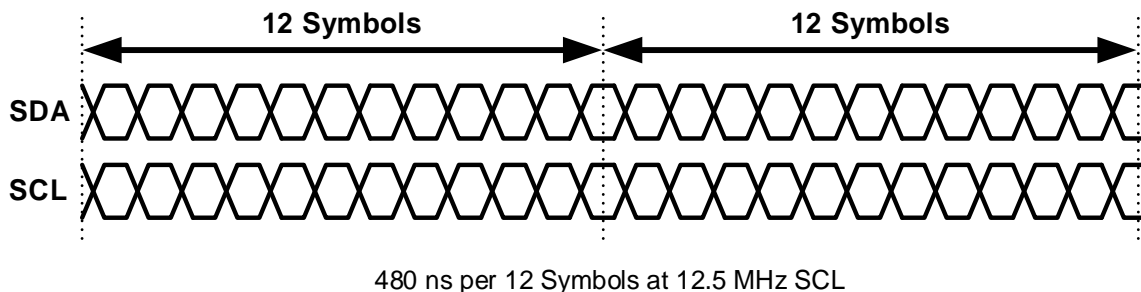
5.2.3.1.2 Ternary Coding Protocol

2804 The HDR Ternary Modes use Ternary Coding Protocol. Ternary Coding Protocol handles two bytes at a time
 2805 (packed in 16 bits), adds two Parity bits, and encodes the resulting 18 bits into a set of twelve Ternary
 2806 Symbols. Each Symbol is transmitted as a level transition of SCL line, a level transition of the SDA line, or
 2807 simultaneous level transitions on both lines. This method uses both lines for data, instead of one clock line
 2808 and one data line.

2809 At the receiver side, each level change on SCL, SDA, or both, is converted back to a Ternary Symbol. These
 2810 Symbols are grouped and then decoded into 18 bits; after Parity checking the original 16 data bits have been
 2811 recovered and become available for use as the original two data bytes.

2812 For HDR-TSP Mode, each data Word (16 data bits plus two Parity bits) is transmitted on the I3C Bus as a
 2813 run of 12 Ternary Symbols, again using both lines for data. **Figure 54** illustrates I3C Bus transmission of
 2814 these 12-Symbol runs.

2815 For HDR-TSL Mode, the waveforms are the same except that the number of Symbols and the Frame duration
 2816 both increases.



2817
2818

Figure 54 Twelve Symbols Per Data Word

2820 Each 12-Symbol run is treated as six Symbol Pairs, transmitted with the most significant Symbol Pair first.
 2821 Within each Symbol Pair the most significant Symbol is sent first. The two data bytes are transmitted in the
 2822 following order:

- 2823 1. The first data byte, starting with the MSb and progressing down to the LSb
- 2824 2. The second data byte, starting with the MSb and progressing down through the LSb
- 2825 3. The two parity bits, P1 and P0.

Ternary Encoding

2826 The coding of the 16 data bits is done in seven phases:

2827 1. Calculate the two Parity bits from the 16 data bits (two bytes) as shown in **Table 66**:

2828 P1 shall contain the XOR of all the odd-index data bits

2829 P0 shall contain the XOR of all the even-index data bits

2830

Table 66 Parity Bits

Parity Bit	Works with Data Bits	Notes
P1	15, 13, 11, 9, 7, 5, 3, 1	P1 bit is XOR of all the odd index data bits
P0	14, 12, 10, 8, 6, 4, 2, 0	P0 bit is XOR of all the even index data bits

2831 2. Form the 18-bit string, by appending the two parity bits at the LSb side of the 16-bit data.

2832 3. Parse the 18-bit string out into six 3-bit triplets

2833 4. For each 3-bit triplet, determine the needed Symbol Pair. Use either **Table 67**, or the following
2834 formula:

2835

$$B2 * 2^2 + B1 * 2^1 + B0 * 2^0 = T1 * 3^1 + T0 * 3^0$$

2836

Where:

2837 · B2, B1 and B0 are the values (0 or 1) of the high, middle, and low bits respectively (binary
2838 coefficients) of the triplet

2839 · T1 and T0 are the values (0, 1, or 2) of the first and second Symbols respectively in the
2840 resulting Symbol Pair

2841

Table 67 Converting 3-bit Binary Triplet Value to Ternary Symbol Pair

Binary Triplet Value				Equivalent Symbol Pair		
B2	B1	B0	Decimal	First Symbol (T1)	Second Symbol (T0)	Symbol Pair
0	0	0	0	0	0	0,0
0	0	1	1	0	1	0,1
0	1	0	2	0	2	0,2
0	1	1	3	1	0	1,0
1	0	0	4	1	1	1,1
1	0	1	5	1	2	1,2
1	1	0	6	2	0	2,0
1	1	1	7	2	1	2,1
(not applicable)			Special: 8 is an error	2	2	2,2

2842 **Symbol Pair Encoding Example**2843 **Method 1: Using Table 67**

- 2844 · Triplet to be encoded: 3'b 101 or decimal 5
 2845 · Resulting Symbol Pair: 2'T 12, i.e. T1 = 1 and T0 = 2

2846 **Method 2: Using Formula**

- 2847 · Start with the formula:

$$2849 \quad B2 * 2^2 + B1 * 2^1 + B0 * 2^0 = T1 * 3^1 + T0 * 3^0$$

2848 Substitute the triplet bit values for B2, B1, and B0, and determine the equation's value:

$$2850 \quad 1 * 2^2 + 0 * 2^1 + 1 * 2^0 = ?$$

$$2851 \quad 1 * 4 + 0 * 2 + 1 * 1 = ?$$

$$2852 \quad 4 + 0 + 1 = 5$$

- 2853 · Determine the necessary values for the two Symbols (ternary coefficients) in the Symbol Pair,
 2854 T1 and T0:

$$2855 \quad 5 = T1 * 3^1 + T0 * 3^0$$

$$2856 \quad 5 = T1 * 3 + T0 * 1$$

$$2857 \quad 5 = 1 * 3 + 2 * 1$$

$$2858 \quad T1 = 1, T0 = 2$$

- 2859 · Express the value of each Symbol (ternary coefficient) as 2 bits. **Table 68** shows which line or
 2860 lines must change level during a time period in order to transmit a given Symbol. If the high
 2861 bit of the Symbol value is 0, then the SCL line must change level. If the low bit of the Symbol
 2862 value is 0, then the SDA line must change level. **Figure 55** shows the transitions.

2863 **Table 68 Converting Ternary Symbol Value to SCL and SDA Level Changes**

Ternary Symbol Value (Ternary Coefficient)		SCL and SDA Level Changes		
Ternary	Binary	SCL,SDA	SCL Change	SDA Change
1'T 0	2'b00	0,0	Yes	Yes
1'T 1	2'b01	0,1	Yes	No
1'T 2	2'b10	1,0	No	Yes
Note: <i>The Symbol Value 2'b11 (which is equal to decimal 3) is not used, because that would indicate that neither SCL nor SDA changed level. In Ternary Signaling, at least one line must change level in order to clock the transmission.</i>				

- 2864 5. For each Symbol (ternary coefficient), determine the necessary level changes of the SCL and SDA
2865 wires during that time slot, using the formulas:

2866
$$\text{Symbol Code Bit 1} = (\text{SCL level}) \text{ XNOR } (\text{Previous SCL level})$$

2867
$$\text{Symbol Code Bit 0} = (\text{SDA level}) \text{ XNOR } (\text{Previous SDA level})$$

2868 **Note:**

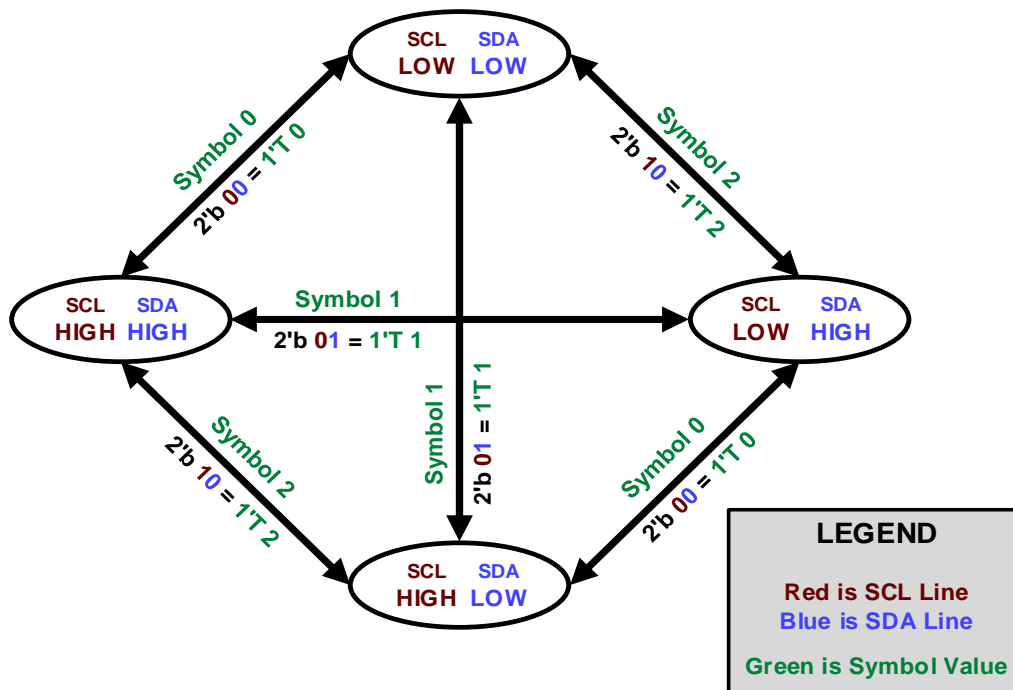
2869 XNOR is the logical Exclusive NOR function. It returns “TRUE” or “1” if there is NO CHANGE between
2870 the present and previous state, otherwise it returns “FALSE” or “0”.

- 2871 6. Concatenate the results for each triplet in the corresponding order.

2872 For HDR-TSP the resultant string of SCL and SDA line level changes shall be directly applied to
2873 the SCL and SDA lines.

- 2874 7. For HDR-TSL, an additional encoding step shall be applied. If a change to the SCL line results in
2875 the High state, and if the value of the following legitimate Symbol is not 1'T 0 (2'b00), then the
2876 encoder shall insert an additional “dummy” Symbol with value 1'T 1 (2'b01) (i.e. the transmitter
2877 shall pull the SCL line Low while keeping the SDA line unchanged). Decoders shall ignore this
2878 “dummy” Symbol.

2879 *Figure 55* is a state diagram illustrating the four possible states (combinations of High and Low levels on
2880 SCL and SDA), and how reception of the three possible Ternary Symbols (0, 1, and 2) causes transitions
2881 from one state to the next.



2882
2883

Figure 55 SCL and SDA State Transitions and Their Symbolic Coding

2884 **16-Bit Ternary Encoding Example**

2885 Assume that the 16-bit value to be encoded is: 0x852B or 16'b 1000 0101 0010 1011.

- 2886 · Determine the values of Parity bits P1 and P0: 1'b0 and 1'b1
- 2887 · Form an 18-bit string by appending P1 and P0 to the 16-bit value: 18'b 1000 0101 0010 1011 01
- 2888 · Regard the 18 bits as six 3-bit triplets: 18'b 100 001 010 010 101 101
- 2889 · For each triplet, determine the needed Symbol Pair:
- 2890 · 3'b 100 à 2'T 11
- 2891 · 3'b 001 à 2'T 01
- 2892 · 3'b 010 à 2'T 02
- 2893 · 3'b 010 à 2'T 02
- 2894 · 3'b 101 à 2'T 12
- 2895 · 3'b 101 à 2'T 12
- 2896 · Grouping the six Symbol Pairs, we have: 12'T 11 01 02 02 12 12

2897 The Symbols to be transmitted, and the resulting SCL and SDA transition states to be transmitted,
2898 depend upon which HDR Ternary Mode is being used (see **Figure 56**):

- 2899 · For HDR-TSP, 12 Symbols would be transmitted. The SCL and SDA states would be:
- 2900 01 01 00 01 00 10 00 10 01 10 01 10
- 2901 · For HDR-TSL, 16 Symbols would be transmitted. The SCL and SDA states would be:
- 2902 01 **01** 01 00 01 00 10 00 **01** 10 01 **01** 10 01 **01** 10
- 2903 For HDR-TSL only, the additional “dummy” Symbols shown in **bold italics** are inserted.

Ternary Decoding

2906 On the Decoder side, the received Symbols are decoded into an 18-bit string (16 data bits and two Parity bits)
2907 using a five step process:

2908 1. Twelve Symbols are received. Each received Symbol is treated as a 2-bit Symbol Value, following
2909 the reverse of the procedure described in the Encoding section. The level changes on the SCL and
2910 SDA lines are evaluated and the Ternary Symbols are recovered, as per **Table 68**:

- 2911 · Symbol Value Bit 1 = (SCL level) XNOR (Previous SCL level)
- 2912 · Symbol Value Bit 0 = (SDA level) XNOR (Previous SDA level)

2913 This process is different, depending upon which HDR Ternary Mode is being used:

- 2914 a. For HDR-TSP Mode, recover sets of 12 Symbols (6 Symbol Pairs) and use them all.
- 2915 b. For HDR-TSL Mode, an additional decoding step is needed: Discard every 2'b01 Symbol if
2916 the SCL line is Low. The Decoder continues with this procedure until 12 legitimate Symbols
2917 (6 Symbol Pairs) are recovered.

2918 2. Adjacent Symbols are treated as Symbol Pairs, for a total of six Symbol Pairs. Each Symbol Pair
2919 may be regarded as a two-digit ternary numbers with digits T1 (3's place) and T0 (1's place).

2920 3. Each Symbol Pair's value is converted to one 3-bit triplet, using either **Table 67** or the following
2921 formula:

$$2922 \quad T1 * 3^1 + T0 * 3^0 = B2 * 2^2 + B1 * 2^1 + B0 * 2^0$$

2923 Where T1 and T0 are the values (0, 1, or 2) of the Symbol Pair's two ternary digits, and B2, B1,
2924 and B0 are the values (0 or 1) of the bits in the resulting triplet.

2925 4. The six triplets are concatenated to a single 18-bit value. This includes both the original 16-bit
2926 data byte pair, and the two parity bits P0 and P1.

2927 5. The two Parity bits are verified and then discarded. At this point, the original byte pair has been
2928 recovered.

2929 16-Bit Ternary Decoding Example

2930 Assume that the recovered Symbols (ternary coefficients) are 12'T 110102021212.

2931 · Regarded as Symbol Pairs, they are: 12'T 11 01 02 02 12 12

2932 · For each Symbol Pair, determine the needed binary triplet:

2933 · 2'T 11 à 3'b 100

2934 · 2'T 01 à 3'b 001

2935 · 2'T 02 à 3'b 010

2936 · 2'T 02 à 3'b 010

2937 · 2'T 12 à 3'b 101

2938 · 2'T 12 à 3'b 101

2939 · Concatenate the triplets into a single binary number: 18'b 100 001 010 010 101 101

2940 · Regarded as a 16-bit number plus two trailing Parity bits, this is: 18'b 1000 0101 0010 1011 01

2941 · Discarding the Parity bits (after checking them), the fully decoded data is 16'b 1000 0101 0010
2942 1011, or 0x85 and 0x2B.

2943 Parity error handling is further specified at **Section 5.2.3.4**.

5.2.3.2 HDR Ternary Command Coding

2944 In the HDR Ternary Modes, a Command Word follows the initial Enter HDR CCC (and any HDR Restart).
2945 This Command Word uses the normal 18-bit model.

2946 **Table 69** illustrates the Command Word format for the HDR Ternary Modes.

2947 **Table 69 HDR Ternary Command Word Format**

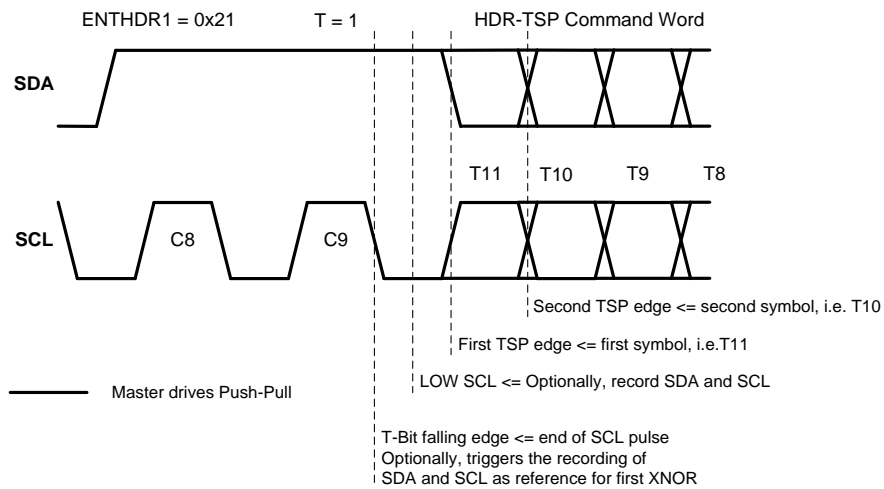
Bits	Field	Size (bits)	Notes
15	Read / Write	1	1=Read (Slave to Master) 0=Write (Master to Slave)
14:8	Command Code	7	128 possible Write Commands, 128 possible Read Commands
7:1	Slave Address	7	Same Dynamic Address as used in I3C SDR Protocol
0	Reserved	1	Reserved for future use

2948 Regarding bits 15:8 together, the possible Command Code spaces for Read Commands and Write Commands
2949 in HDR Ternary Modes are illustrated in **Table 70**.

2950 **Table 70 Read and Write Command Spaces for HDR Ternary Modes**

Command Codes	Command Functions
0x00 – 0x7F	128 possible Write Commands
0x80 – 0xFF	128 possible Read Commands

2951 **Figure 57** is the signal diagram for starting the HDR-TSP Command Code after entering HDR-TSP Mode.
2952 HDR-TSL mode is similar.



2953

2954 **Figure 57 HDR-TSP Command Code After ENTHDR1**

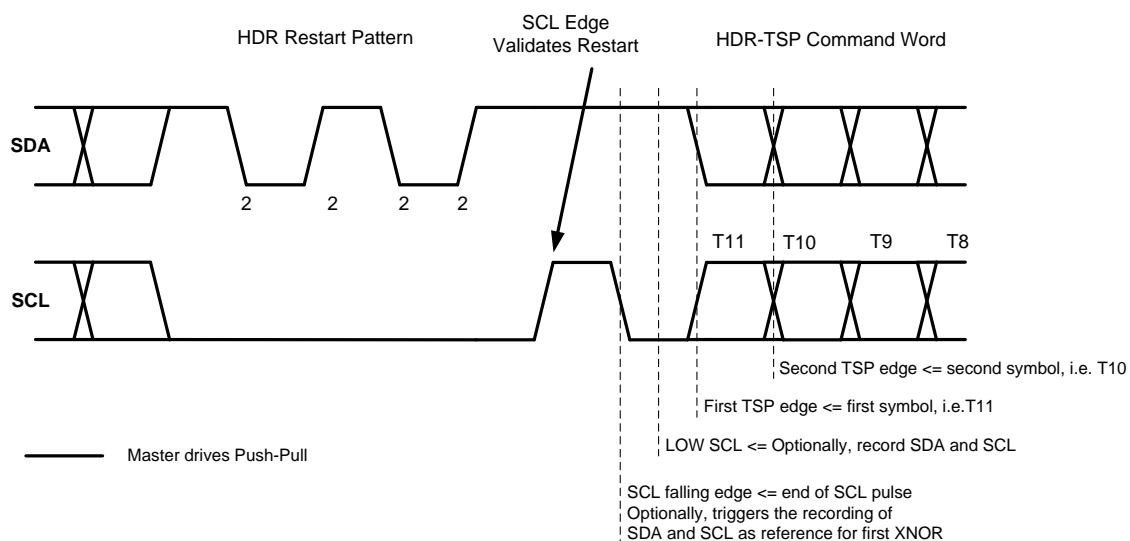
2955 As **Figure 57** shows:

- 2956 1. The first SDA or SCL change can take place only after one symbol duration elapses after the SCL
2957 pulse falling edge (i.e., half a clock cycle). This time must be at least 32ns, as per **Table 75**.
2958 2. The first level change of SDA, SCL, or both signals the first ternary symbol, i.e. T11.

- 2959 3. In order to determine the value of T11, the Slave calculates the XNOR between the present and
 2960 previous levels of SDA and of SCL. The Slave can obtain the needed previous SDA and SCL
 2961 levels in either of two ways.
- 2962 The Slave can either:
- 2963 a. Preserve the levels (SDA=1 and SCL=0), as in the ENTHDR1 procedure; or
 - 2964 b. When triggered by the falling edge of the T-Bit SCL pulse, sample the values of SDA and
 2965 SCL in the same manner as it would for the normal HDR-TSP Mode, and store the values as
 2966 references for the first XNOR.

2967 Option b is better than option a.

2968 **Figure 58** is the signal diagram for starting the HDR-TSP Command Code after the HDR Repeated Start
 2969 Pattern. HDR-TSL mode is similar.



2970

2971

Figure 58 HDR-TSP Command Code After HDR Repeated Start Pattern

2972 As **Figure 58** shows:

- 2973 1. After the SCL rising edge (which validates the HDR Repeated pattern), an SCL falling edge is
 2974 provided. The resulting SCL pulse is identical to the SCL Pulse of the T-Bit (see **Figure 57**).
- 2975 2. The first SDA or SCL change can take place only after one symbol duration elapses after the SCL
 2976 pulse falling edge (i.e., half a clock cycle). This time must be at least 32 ns, per **Table 75**.
- 2977 3. The first level change of SDA, SCL, or both signals the first ternary symbol, i.e. T11.
- 2978 4. In order to determine the value of T11, the Slave calculates the XNOR between the present and
 2979 previous levels of SDA and of SCL. The Slave can obtain the needed previous SDA and SCL
 2980 levels in either of two ways.

2981 The Slave can either:

- 2982 a. Preserve the levels (SDA=1 and SCL=0) as eventually expected and recorded immediately
 2983 after the falling edge of the validating SCL pulse, or
- 2984 b. When triggered by the falling edge of the validating SCL pulse, sample SDA and SCL in the
 2985 same manner as it would for the normal HDR-TSP Mode, and store the values as references
 2986 for the first XNOR.

2987 Option b is better than option a.

5.2.3.3 HDR Ternary Bus Turnaround

2988 When the HDR Master sends a Read Command while in HDR-TSP Mode or HDR-TSL Mode, the I3C Bus
2989 shall turnaround in order to allow the HDR Slave to transmit its Symbols. Likewise, when the HDR Slave is
2990 finished sending its data, the I3C Bus shall turnaround in order to allow the HDR Master to regain Bus
2991 control.

2992 The first Data Word from the Slave after a Read Command is accomplished by the Master Parking the I3C
2993 Bus High (SCL High and SDA High) in one edge, and then tri-stating. The Slave then drives the next edge.
2994 The Slave may take many hundreds of nanoseconds to start, if needed.

2995 If the Slave wishes to NACK the Read, then it shall still take over the line, but it shall then immediately
2996 transmit the start of an HDR Restart Pattern or HDR Exit Pattern, emitting no Data Words, just as it would
2997 after normal Data (see below).

2998 The end of the Slave Read Message is signaled by the Slave transmitting the start of the HDR Restart Pattern
2999 and HDR Exit Pattern. This signaling shall be performed only after a completed Data Word, or when no Word
3000 is to be emitted (NACKed). This signaling is done via the following steps:

- 3001 1. The Slave transmits a (non-data) Symbol that sets SCL Low and SDA High. (If SCL is already in
3002 the Low state, and SDA is already in the High state from the end of the last data, then this step is
3003 skipped.)
- 3004 2. The Slave transmits the Symbol 2 three times in a row. The first Symbol 2 drives SDA Low (while
3005 SCL stays Low), then the second Symbol 2 drives SDA High (while SCL stays Low), and then the
3006 third Symbol 2 drives SDA Low again (while SCL stays Low). Since three Symbol 2s in a row is
3007 not a legal data coding, the Master shall recognize this as a handover at end of the third Symbol 2.
- 3008 3. The Slave then releases both SCL and SDA to High-Z. This release shall use normal release
3009 timing; i.e., the duration of the release shall be well less than the minimum defined in *Section 6*
3010 (see *Table 76*).
- 3011 4. As soon as the Master sees SDA go Low in this condition (i.e., after three Symbol 2s in a row,
3012 leaving both SCL and SDA Low), the Master shall then begin asserting both SCL Low and SDA
3013 Low. This assertion shall use the Master's normal Low assertion timing.

3014 **Note:**

3015 *For some period of time, the Master and the Slave will both be driving SCL Low, and also both be*
3016 *driving SDA Low. This is both valid and safe.*

- 3017 5. The Master shall then wait for a period of at least the minimum t_{Edge} defined in *Section 6* (see
3018 *Table 76*), and then finally finish the HDR Restart Pattern or HDR Exit Pattern.

5.2.3.4 HDR Ternary Error Detection

3019 Two error types are defined for the HDR Ternary Modes:

- 3020 1. More than one Symbol 2 in a row. (In Symbol 2 the SCL line does not change, and the SDA line
3021 does change.)
- 3022 · Exception: More than one Symbol 2 in a row is allowed for HDR Restart or HDR Exit, but only
3023 from a known starting state (SCL Low and SDA High), and only at a Data Word boundary
3024 (allowing for one Symbol to set up that state). Thus the Symbol pattern 2,2 is only allowed under
3025 the following situations:
 - 3026 · As part of the HDR Exit Pattern, which uses 3 or 4 such Symbol pairs and involves SCL Low
 - 3027 · As part of the HDR Restart pattern, which uses 2 such Symbol pairs and involves SCL Low
 - 3028 · As the result of one normal Symbol ending in 2, and the next Symbol starting with 2.
- 3029 2. Parity errors (see *Section 5.2.3.1.2*).

3030 An error may be seen by an HDR Slave, or by an HDR Master when a Slave is driving the I3C Bus:

- 3031 · A Slave that sees an error shall stop tracking Symbols and use HDR Exit and HDR Restart Pattern
3032 detection. This means to wait until safe. A Slave shall enable the HDR Exit and HDR Restart
3033 Pattern Detector upon entry to any HDR Mode, ensuring that it is always safe.
- 3034 · A Master that sees an error shall wait until the Slave stops transitioning the Bus for 2x the
3035 maximum edge-to-edge duration used by that Slave. Then the Master shall force out the HDR Exit
3036 Pattern.

3037 If an error occurs in the RnW Bit in a Private Read Transfer, then the Write Data from the Master might
3038 conflict with the Read Data from the Slave. Both the Master and the Slave should always monitor the Data
3039 they transmit. If the Master or Slave performs such monitoring, and if the monitored Data differs from the
3040 Data the Master or Slave intended to send (except for data transferred during the Dynamic Address
3041 Arbitration procedure), then this shall be considered an error. After detecting this error, both the Master and
3042 the Slave shall stop the transmission and then the Slave shall wait for HDR EXIT or HDR Restart. After the
3043 Master sends HDR EXIT, it may retry the transmission.

3044 **Note:**

3045 *It could be appropriate to wait for the Symbol 2,2, but the Master would still need to see quiescence.*
3046 *If the Master is seeing errors, then the results of even a Symbol 2,2 are questionable.*

6 I3C Electrical Specifications

6.1 DC I/O Characteristics

3047 This Section describes the DC operating parameters of the I3C interface in two modes: Push-Pull Mode and
3048 Open Drain Mode. In Push-Pull Mode, the SDA pin drives at higher speeds with a totem-pole driver.

3049 Two important parameters should be noted for I3C:

- 3050 · The I3C interface targets nominal operating voltages of 1.2V, 1.8V, and 3.3V or less. The I3C
3051 interface is not characterized for 5V systems, but could be extended to support 5V if sufficient
3052 driver strength, reduced diameter, and/or reduced speed is used in the system.
- 3053 · For peak speeds the capacitance loading allowed is reduced to 50pF, which is much lower than
3054 Legacy I²C. With I3C higher capacitance busses are possible, but only at reduced speeds and with
3055 reduced features (e.g. no I²C Devices are supported).

3056

Table 71 I3C I/O Stage Characteristics Common to Push-Pull Mode and Open Drain Mode

Parameter	Symbol	Conditions	Min	Typ	Max	Unit	Notes
Operating Voltage	V _{DD}	–	1.10	1.20	1.30	V	1
			1.65	1.80	1.95		
			2.97	3.30	3.63		
Low-Level Input Voltage	V _{IL}	–	-0.3	–	0.3 * V _{DD}	V	
High-Level Input Voltage	V _{IH}	–	0.7 * V _{DD}	–	V _{DD} + 0.3	V	
Schmitt Trigger Inputs Hysteresis	V _{hys}	–	0.1 * V _{DD}	–	–	V	
Output Low Level	V _{OL}	For V _{DD} < 1.4V: I _{OL} = 2 mA	–	–	0.18	V	2
		For V _{DD} ≥ 1.4V: I _{OL} = 3 mA	–	–	0.27	V	2
Input Current (per Input-Only I/O Pin)	I _i	-100 mV < V _i < V _{DD} + 100 mV for ≥ 1.8V nominal	-10	–	10	μA	2
		-100 mV < V _i < V _{DD} + 100 mV for < 1.8V nominal	-5	–	5	μA	2
Capacitance (per I/O Pin)	C _i	For < 1.8V nominal	–	–	5	pF	5
		For ≥ 1.8V nominal	–	–	10	pF	5
Capacitance Mismatch Between Pins	ΔC	Difference between SDA and SCL capacitance C _i ≤ 5pF	–	–	1.5	pF	5
		Difference between SDA and SCL capacitance C _i > 5pF	–	–	3	pF	5
Push-Pull Only							
Output High Level	V _{OH}	For V _{DD} < 1.4V: I _{OH} = -2 mA	V _{DD} - 0.18	–	–	V	2
		For V _{DD} ≥ 1.4V: I _{OH} = -3 mA	V _{DD} - 0.27	–	–	V	2
Legacy Mode with Pull-Up							
Pull-Up for Open Drain	R _p	t _r = Max rise time C _b = Bus Capacitance V _{DD} = 1.2V, 1.8V, or 3.3V	$\frac{V_{DD} - V_{OL}}{3 \text{ mA}}$ for V _{DD} ≥ 1.4V	$\frac{t_r}{0.8473 * C_b}$	Ω	3, 4, 6, 7	
		t _r = 120 ns C _b = 50 pF	$\frac{V_{DD} - V_{OL}}{2 \text{ mA}}$ for V _{DD} < 1.4V	2833			

Note:

- 1) *This Specification considered only 1.2V, 1.8V, and 3.3V, with associated tolerances. Other voltage ranges are not prohibited. Any I3C Bus implementation shall ensure the correct operation of the Devices resident on the Bus, notably the inter-compatibility of their voltage ranges.*
- 2) *Negative sign for currents indicates current direction.*
- 3) $V(t1) = 0.3 * V_{DD} = V_{DD} (1 - e^{-t1/RC})$; then $t1 = 0.3566749 * RC$
 $V(t2) = 0.7 * V_{DD} = V_{DD} (1 - e^{-t2/RC})$; then $t2 = 1.2039729 * RC$
 $T = t2 - t1 = 0.8473 * RC$
- 4) *Pull-Up for Open Drain shall be switched off during I3C Push-Pull operation. May be implemented as: A Pull-Up internally; A current source internally; or an external Pull-Up resistor driven by a pin.*
- 5) *For Devices that support both 1.8V and 3.3V, the larger capacitance may be used.*
- 6) *Open-Drain never occurs on SCL*
- 7) *A weak pull-up needs to be applied as well for Master handoff*

3057 I3C supports Legacy I²C Slaves. An important feature of an I²C Slave is the 50ns Spike Filter on the SDA
 3058 and SCL pads. If a Spike Filter is implemented on all I²C Devices present on the I3C Bus, then the I3C Bus
 3059 may operate at maximum rated clock frequency. If any I²C Device does not have a Spike Filter, then the I3C
 3060 Bus speed is determined by the slowest Legacy I²C Device without a Spike Filter. Other requirements of an
 3061 I²C Legacy Slave are listed in *Table 72*.

3062 **Table 72 Legacy I²C Device Requirements When Operating on I3C**

Feature	Required	Desirable	Not Used	Not Allowed	Notes
Fm Speed	X	–	–	–	–
Fm+ Speed	–	X	–	–	–
HS and UFm	–	–	X	–	2
Static I ² C Address	X	–	–	–	–
50ns Spike Filter	–	X	–	–	1
Clock Stretching by Slave	–	–	–	X	–
I ² C Extended Address (10 bit)	–	–	X	–	2
I3C Reserved Address	–	–	–	X	–
Note:					
1) Lack of Spike Filter will severely degrade Bus performance and eliminate certain I3C Bus features					
2) “Not Used” means that the I3C Master will not make use of the I ² C feature, however if the Slave supports the feature, then it will not interfere with I3C Bus operation.					

6.2 Timing Specification

3063 A key feature of I3C is to maximize the speed of data transfer and minimize the time required for low-power
3064 Devices to remain in Active Mode. In Single Data Rate Mode this is accomplished by keeping Bus
3065 capacitance low and clock speed high. For large packets of data an additional speed improvement is possible
3066 using HDR Modes, where data is transferred on every clock edge. I3C supports Legacy I²C Fm and Fm+
3067 Modes. **Table 73** gives reference timing requirements used in Legacy Mode.

3068

Table 73 I3C Timing Requirements When Communicating With I²C Legacy Devices

Parameter	Symbol	Timing Diagram	Legacy Mode 400kHz / Fm		Legacy Mode 1MHz / Fm+		Units	Notes
			Min	Max	Min	Max		
SCL Clock Frequency	f _{SCL}	–	0	0.4	0	1.0	MHz	–
Setup Time for a Repeated START	t _{SU_STA}	Figure 59	600	–	260	–	ns	–
Hold Time for a (Repeated) START	t _{HD_STA}	Figure 59	600	–	260	–	ns	–
SCL Clock Low Period	t _{LOW}	Figure 59	1300	–	500	–	ns	–
	t _{DIG_L}	Figure 59 Figure 76	t _{LOW} + t _{rCL}	–	t _{LOW} + t _{rCL}	–	ns	–
SCL Clock High Period	t _{HIGH}	Figure 59	600	–	260	–	ns	–
	t _{DIG_H}	Figure 59 Figure 76	t _{HIGH} + t _{rCL}	–	t _{HIGH} + t _{rCL}	–	ns	–
Data Setup Time	t _{SU_DAT}	Figure 59	100	–	50	–	ns	–
Data Hold Time	t _{HD_DAT}	Figure 59	–	–	–	–	ns	–
SCL Signal Rise Time	t _{rCL}	Figure 59	20	300	–	120	ns	–
SCL Signal Fall Time	t _{fCL}	Figure 59	20 * (V _{DD} / 5.5V)	300	20 * (V _{DD} / 5.5V)	120	ns	–
SDA Signal Rise Time	t _{rDA}	Figure 59	20	300	–	120	ns	–
SDA Signal Fall Time	t _{fDA}	Figure 59	20 * (V _{DD} / 5.5V)	300	20 * (V _{DD} / 5.5V)	120	ns	–
Setup Time for STOP	t _{SU_STO}	Figure 59	600	–	260	–	ns	–
Bus Free Time Between a STOP Condition and a START Condition	t _{BUF}	–	1.3	–	0.5	–	μs	–
Pulse Width of Spikes that the Spike Filter Must Suppress	t _{SPIKE}	Figure 76	0	50	0	50	ns	–

3069 During an I3C communication the drive on the SDA pin shall have the ability to dynamically switch between
3070 Push-Pull and Open Drain.

3071

Table 74 I3C Open Drain Timing Parameters

Parameter	Symbol	Timing Diagram	I3C Open Drain Mode		Units	Notes
			Min	Max		
Low Period of SCL Clock	t _{LOW_OD}	Figure 63	200	–	ns	1, 2
	t _{DIG_OD_L}	Figure 63	t _{LOW_ODmin} + t _{fDA_ODmin}	–	ns	–
High Period of SCL Clock	t _{HIGH}	Figure 60	–	41	ns	3, 4
	t _{DIG_H}	Figure 60 Figure 76	–	t _{HIGH} + t _{CF}	ns	–
Fall Time of SDA Signal	t _{fDA_OD}	Figure 63	t _{CF}	12	ns	–
SDA Data Setup Time During Open Drain Mode	t _{SU_OD}	Figure 61 Figure 63	3	–	ns	1
Clock After START (S) Condition	t _{CAS}	Figure 63	38.4 nano	For ENTAS0: 1 μ	seconds	5, 6
				For ENTAS1: 100 μ		
				For ENTAS2: 2 milli		
				For ENTAS3: 50 milli		
Clock Before STOP (P) Condition	t _{CBP}	Figure 64	t _{CASmin} / 2	–	seconds	–
Current Master to Secondary Master Overlap time during handoff	t _{MMOverlap}	Figure 74	t _{DIG_OD_Lmin}	–	ns	–
Bus Available Condition	t _{AVAIL}	–	1	–	μs	7
Bus Idle Condition	t _{IDLE}	–	1	–	ms	–
Time Internal Where New Master Not Driving SDA Low	t _{MMLock}	Figure 74	t _{AVAILmin}	–	us	–

Note:

- 1) This is approximately equal to t_{LOWmin} + t_{DS_ODmin} + t_{rDA_ODtyp} + t_{SU_ODmin}
- 2) The Master may use a shorter Low period if it knows that this is safe, i.e., that SDA is already above V_{IH}
- 3) Based on t_{SPIKE}, rise and fall times, and interconnect
- 4) This maximum High period may be exceeded when the signals can be safely seen by Legacy I²C Devices, and/or in consideration of the interconnect (e.g., a short Bus)
- 5) On a Legacy Bus where I²C Devices need to see Start, the t_{CAS} Min value is further constrained (see Section 5.1.3.5)
- 6) Slaves that do not support the optional ENTASx CCCs (see Section 5.1.9.3.2) shall use the t_{CAS} Max value shown for ENTAS3
- 7) On a Mixed Bus with Fm Legacy I²C Devices, t_{AVAIL} is 300ns shorter than the Fm Bus Free Condition time (t_{BUF})

3072

Table 75 I3C Push-Pull Timing Parameters for SDR and HDR-DDR Modes

Parameter	Symbol	Timing Diagram	Min	Typ	Max	Units	Notes
SCL Clock Frequency	f _{SCL}	–	0.01	12.5	12.9	MHz	1
SCL Clock Low Period	t _{LOW}	Figure 59	24	–	–	ns	–
	t _{DIG_L}	Figure 60	32	–	–	ns	2, 4
SCL Clock High Period for Mixed Bus	t _{HIGH_MIXED}	Figure 60	24	–	–	ns	–
	t _{DIG_H_MIXED}	Figure 60	32	–	45	ns	2, 3
SCL Clock High Period	t _{HIGH}	Figure 59	24	–	–	ns	–
	t _{DIG_H}	Figure 60 Figure 59	32	–	–	ns	2
Clock in to Data Out for Slave	t _{SCO}	Figure 66	–	–	12	ns	–
SCL Clock Rise Time	t _{CR}	Figure 59	–	–	150 * 1 / f _{SCL} (capped at 60)	ns	–
SCL Clock Fall Time	t _{CF}	Figure 59	–	–	150 * 1 / f _{SCL} (capped at 60)	ns	–
SDA Signal Data Hold in Push-Pull Mode	Master	t _{HD_PP}	t _{CR} + 3 and t _{CF} + 3	–	–	–	4
	Slave	t _{HD_PP}	0	–	–	–	–
SDA Signal Data Setup in Push-Pull Mode	t _{SU_PP}	Figure 65 Figure 66	3	–	N/A	ns	–
Clock After Repeated START (Sr)	t _{CASr}	Figure 72	t _{CASmin}	–	N/A	ns	–
Clock Before Repeated START (Sr)	t _{CBSr}	Figure 72	t _{CASmin} / 2	–	N/A	ns	–
Capacitive Load per Bus Line (SDA/SCL)	C _b	–	–	–	50	pF	–
Note:							
1) $F_{SCL} = 1 / (t_{DIG_L} + t_{DIG_H})$							
2) t _{DIG_L} and t _{DIG_H} are the clock Low and High periods as seen at the receiver end of the I3C Bus using V _{IL} and V _{IH} (see Figure 59)							
3) When communicating with an I3C Device on a mixed Bus, the t _{DIG_H_MIXED} period must be constrained in order to make sure that I ² C Devices do not interpret I3C signaling as valid I ² C signaling.							
4) As both edges are used, the hold time needs to be satisfied for the respective edges; i.e., t _{CF} + 3 for falling edge clocks, and t _{CR} + 3 for rising edge clocks.							

3073

Table 76 I3C Push-Pull Timing Parameters for HDR-TSP and HDR-TSL Modes

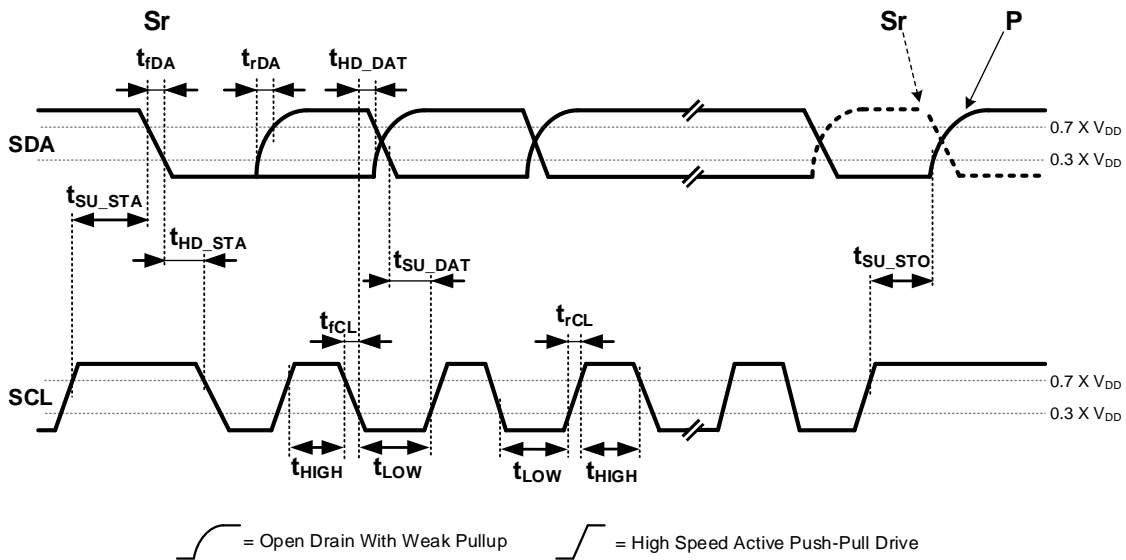
Parameter	Symbol	Timing Diagram	Min	Typ	Max	Units	Notes
Edge-to-Edge Period	t_{EDGE}	Figure 75	t_{DIG_H}	–	–	ns	1, 2
Allowed Difference Between Signals for 'Simultaneous' Change	t_{SKEW}	Figure 75	–	–	12.8	ns	–
Stable Condition Between Symbols	t_{EYE}	Figure 75	12	–	–	ns	–
Time Between Successive Symbols	t_{SYMBOL}	Figure 75	t_{EDGE} Min	–	–	ns	–

Note:

- Edges occur at the rate of $1 / (t_{EDGE} * 2)$
- In a Mixed Bus, HDR-TSL shall respect the maximum $t_{DIG_H_MIXED}$ shown in **Table 75**.

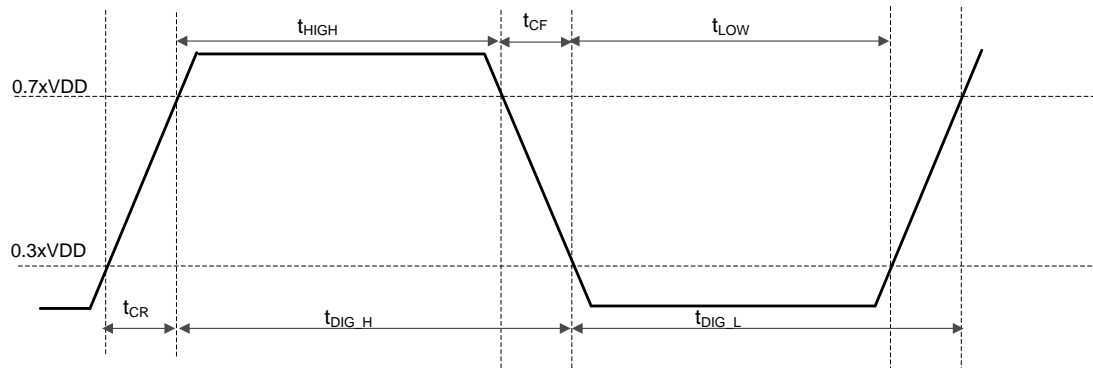
3074
3075

The timing diagram in **Figure 59** depicts I3C Legacy Mode for I²C Devices. The timing parameters referenced in this diagram appear in **Table 73**.



3076
3077
3078

Figure 59 I3C Legacy Mode Timing



3079

3080

Figure 60 tDIG_H and tDIG_L

3081 The start of a typical I3C communication in SDR Mode is shown in **Figure 61**. The initial communication
 3082 looks very similar to I²C, with additional commands that follow as described in **Section 4**. The key difference
 3083 is that higher clock speed is supported, up to 12.5MHz. The higher clock speed allows Legacy I²C Devices
 3084 with 50 ns Spike Filters to ignore the communications. The Master can then run in SDR Mode for I3C Devices
 3085 using full 12.5 MHz timing, though it will have to slow down in order to communicate with Legacy I²C
 3086 Devices. **Figure 61** shows the beginning of a transaction, where the Slave acknowledges its address.

3087 **Figure 62** shows the possible continuation of an I3C SDR communication, in the case where the Slave does
 3088 not acknowledge its address. The Master may either STOP the communication, or else continue with a
 3089 Repeated START.

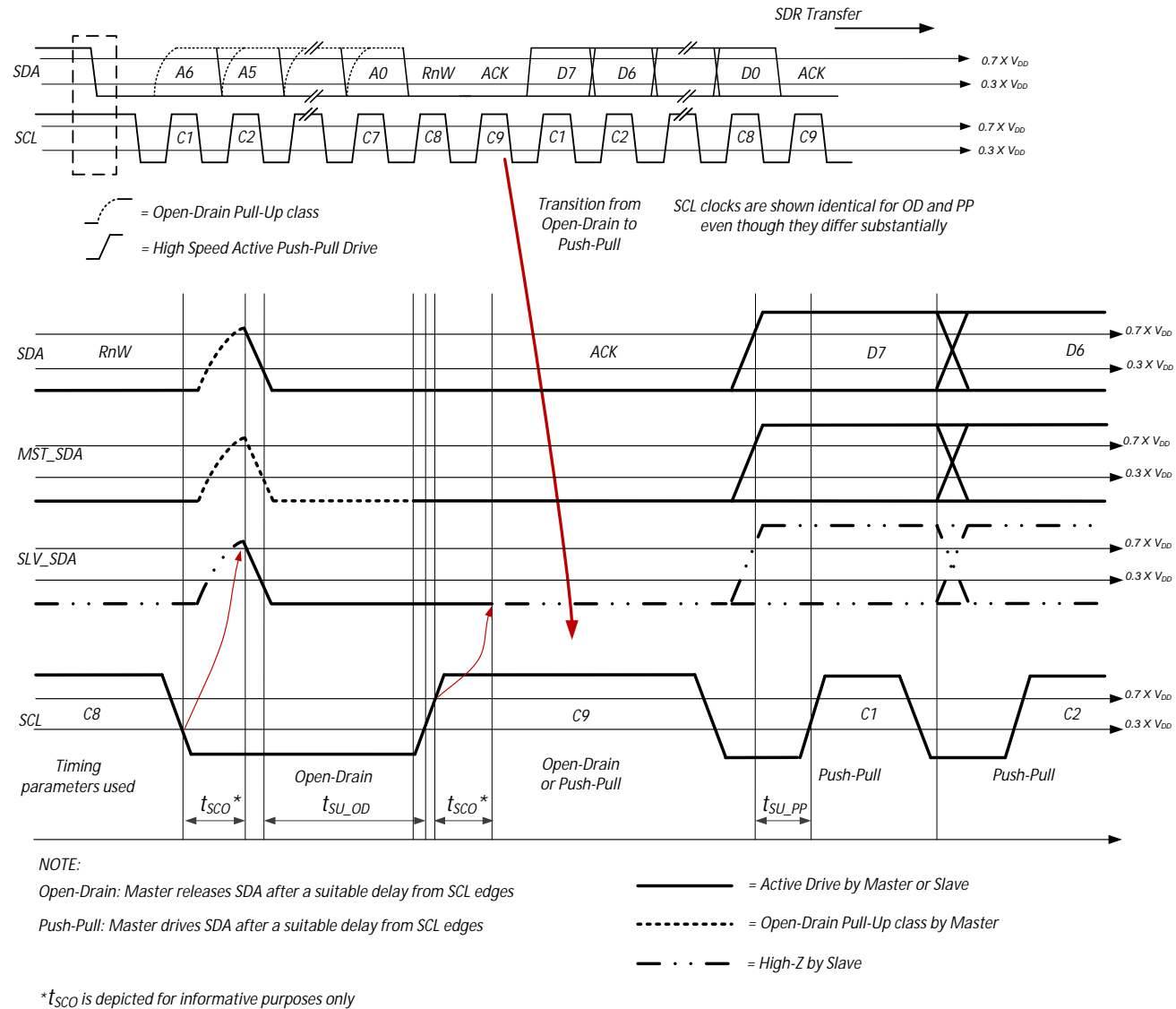
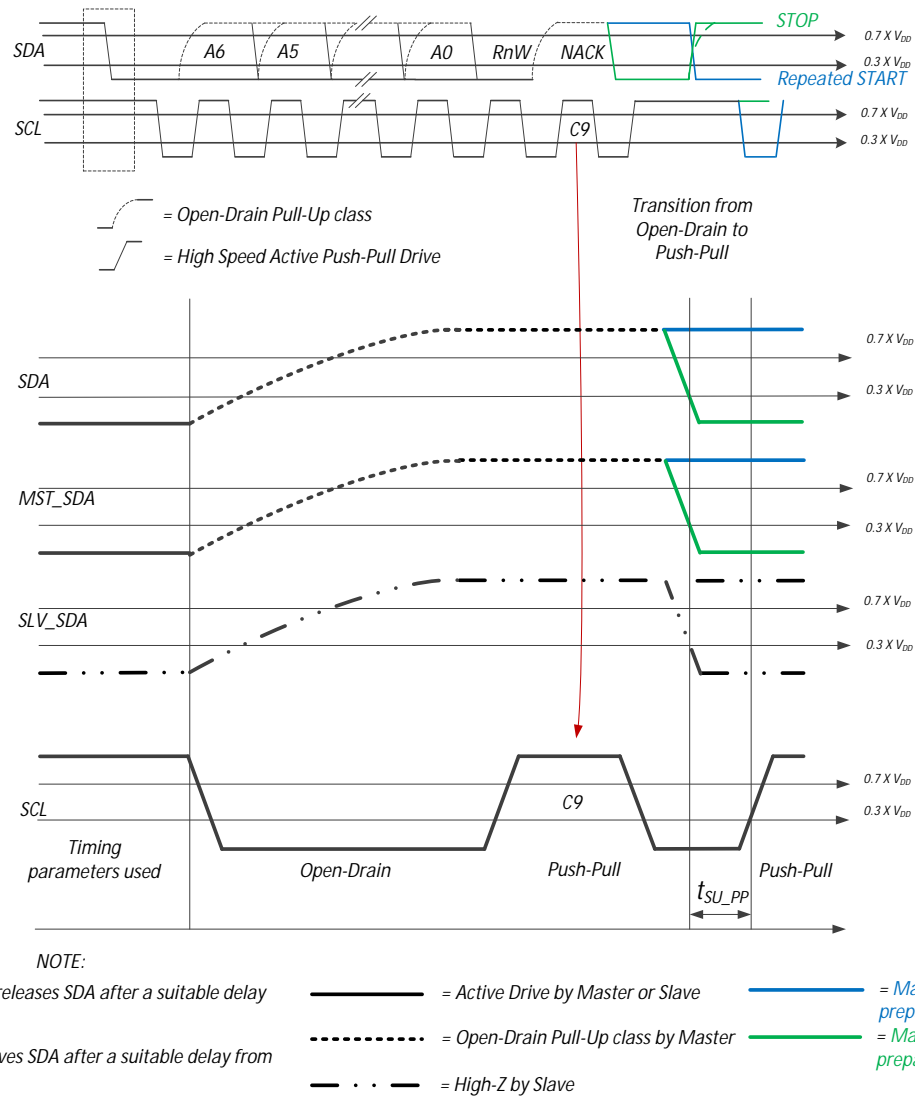


Figure 61 I3C Data Transfer – ACK by Slave

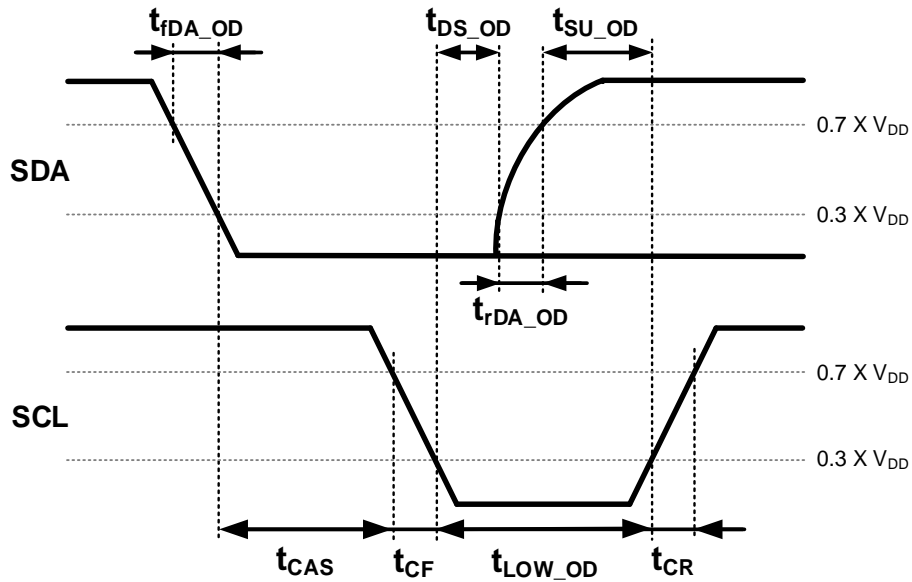
3090
3091



3092
3093

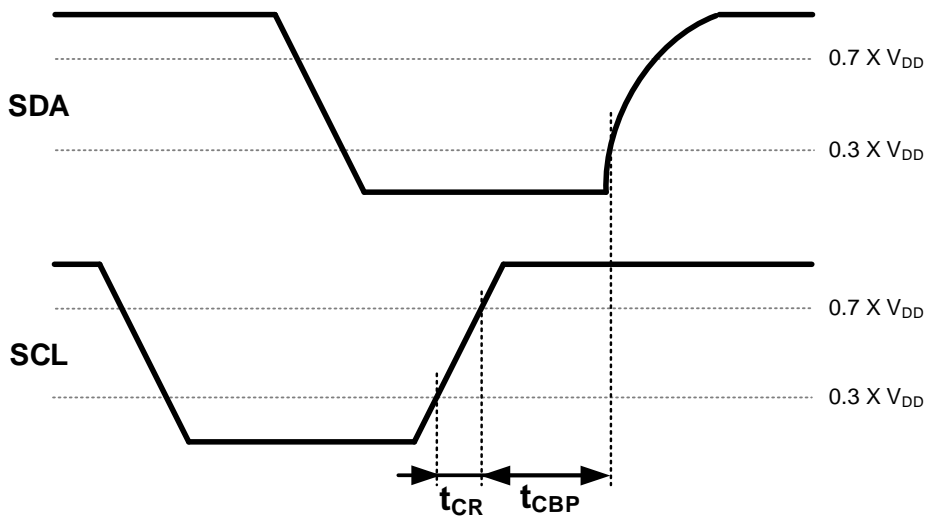
Figure 62 I3C Data Transfer – NACK by Slave

3094 **Figure 63** shows the timing parameters for an I3C START Condition, and **Figure 64** shows the timing
 3095 parameters for an I3C STOP Condition. Notice for both the START and the STOP, the SDA pin is in Open
 3096 Drain mode, as indicated by t_{DV} 's slow rise time.



3097
3098

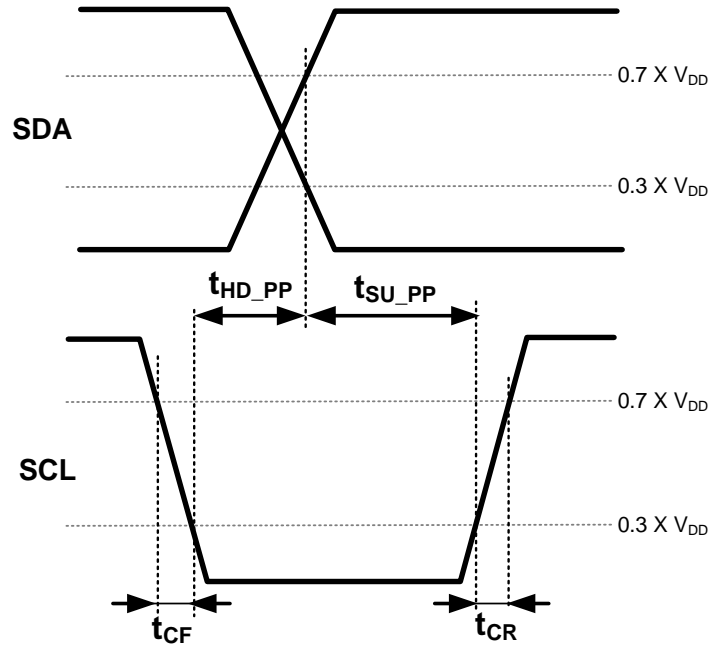
Figure 63 I3C START Condition Timing



3099
3100

Figure 64 I3C STOP Condition Timing

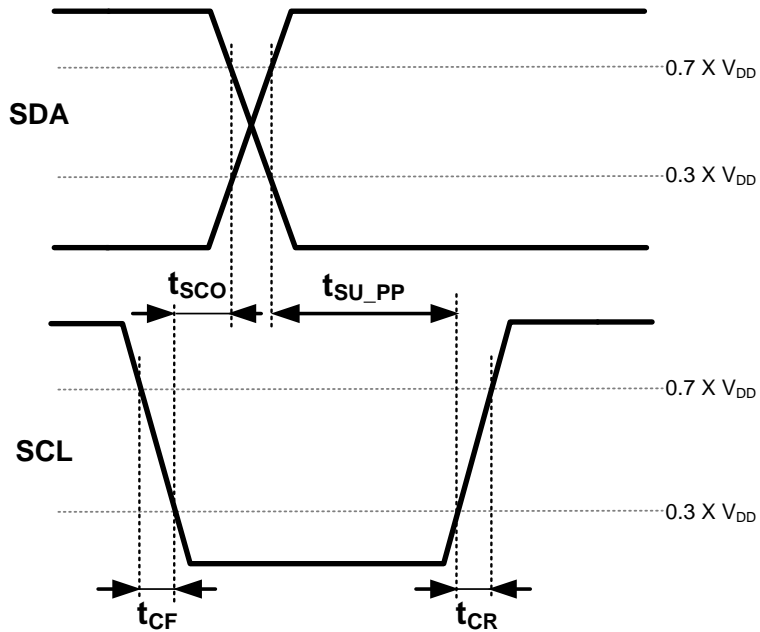
3101 **Figure 65** and **Figure 66** illustrate the timing parameters that are unique to the Master Device and to the
 3102 Slave Device, as specified in **Table 75**. The primary difference between the two is that a Master always
 3103 transmits the clock (SCL), whereas the Slave is receiver-only on the SCL pin.



3104

3105

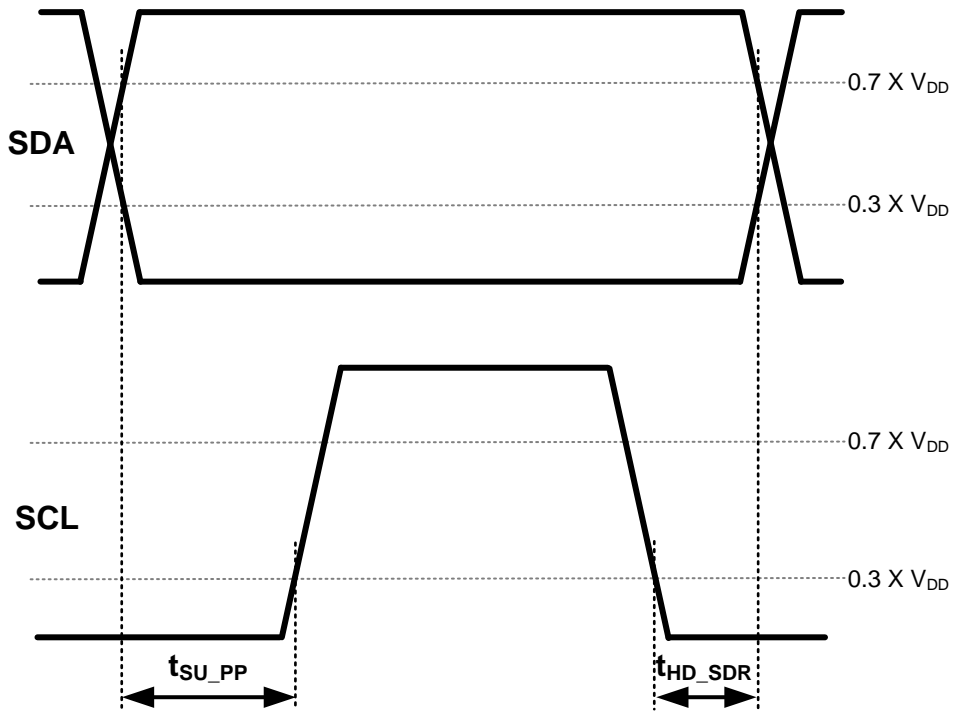
Figure 65 I3C Master Out Timing



3106

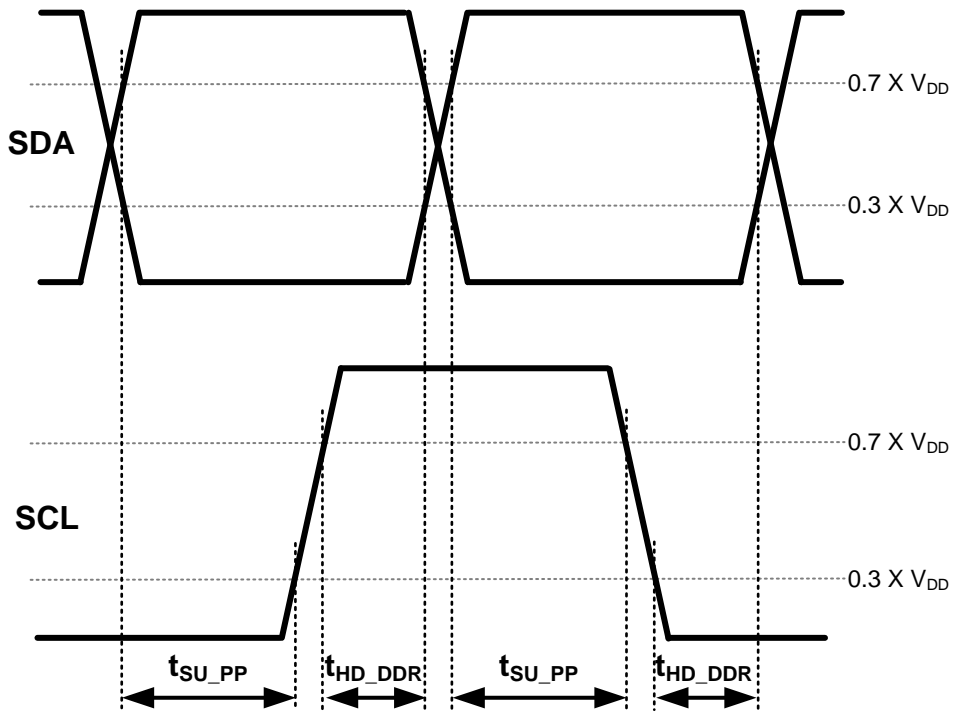
3107

Figure 66 I3C Slave Out Timing



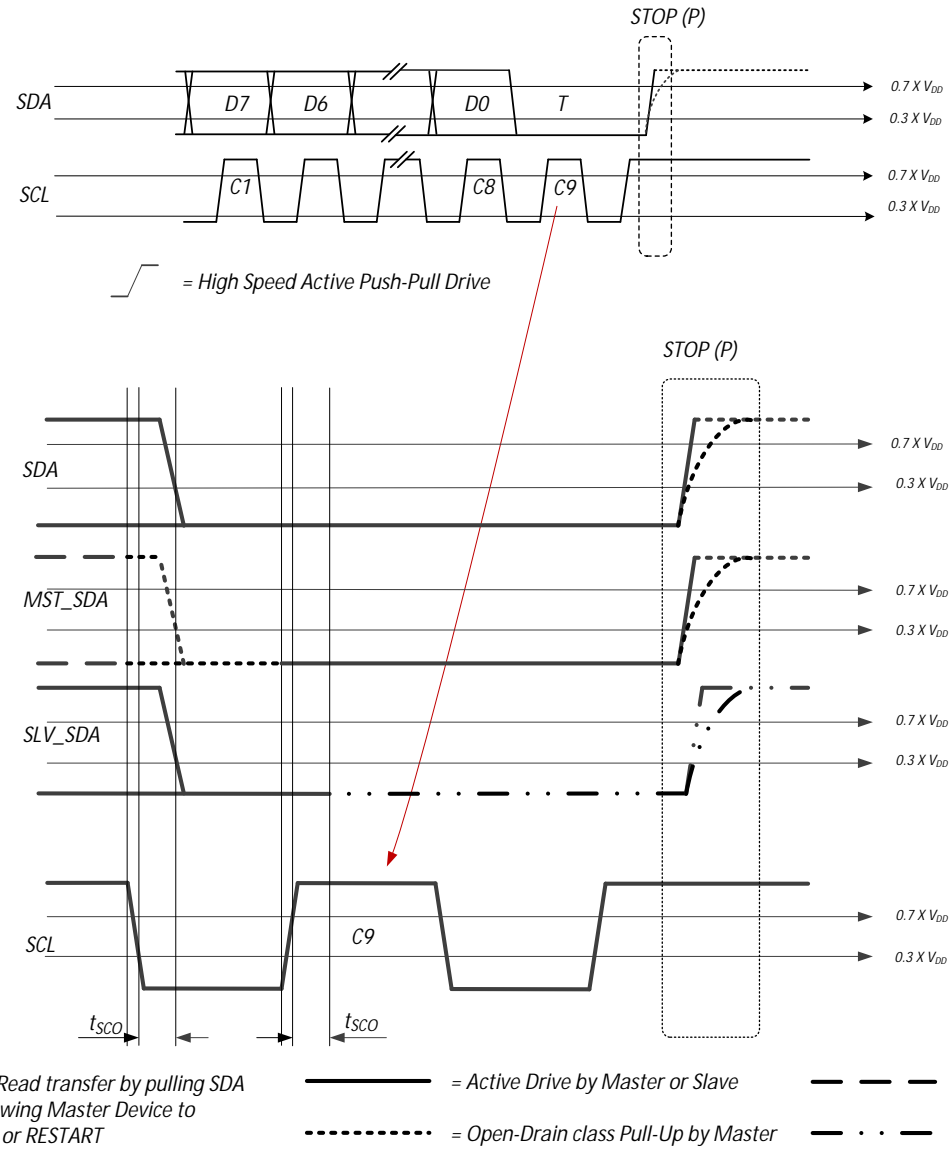
3108
3109

Figure 67 Master SDR Timing



3110
3111

Figure 68 Master DDR Timing

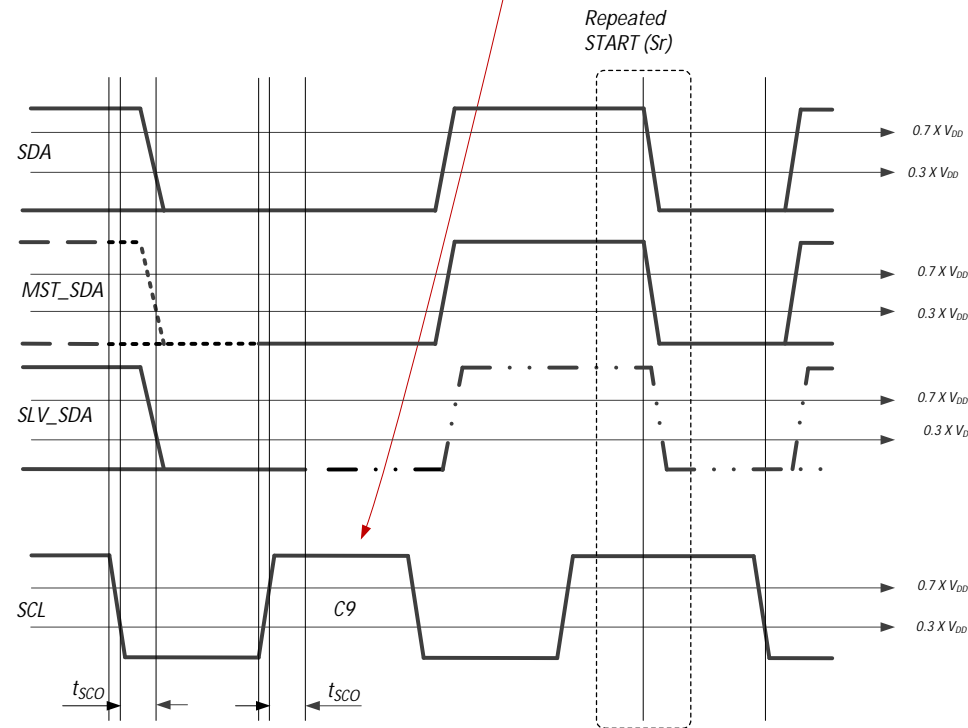
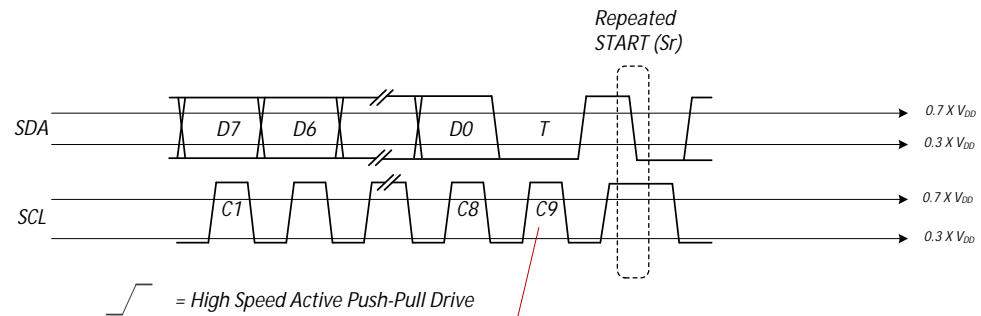


Slave terminates the Read transfer by pulling SDA Low in 9th bit and allowing Master Device to generate either STOP or RESTART

— = Active Drive by Master or Slave
 - - - = High-Z by Master
 ····· = Open-Drain class Pull-Up by Master
 - · - = High-Z by Slave

Figure 69 T-Bit When Slave Ends Read and Master Generates STOP

3112
3113

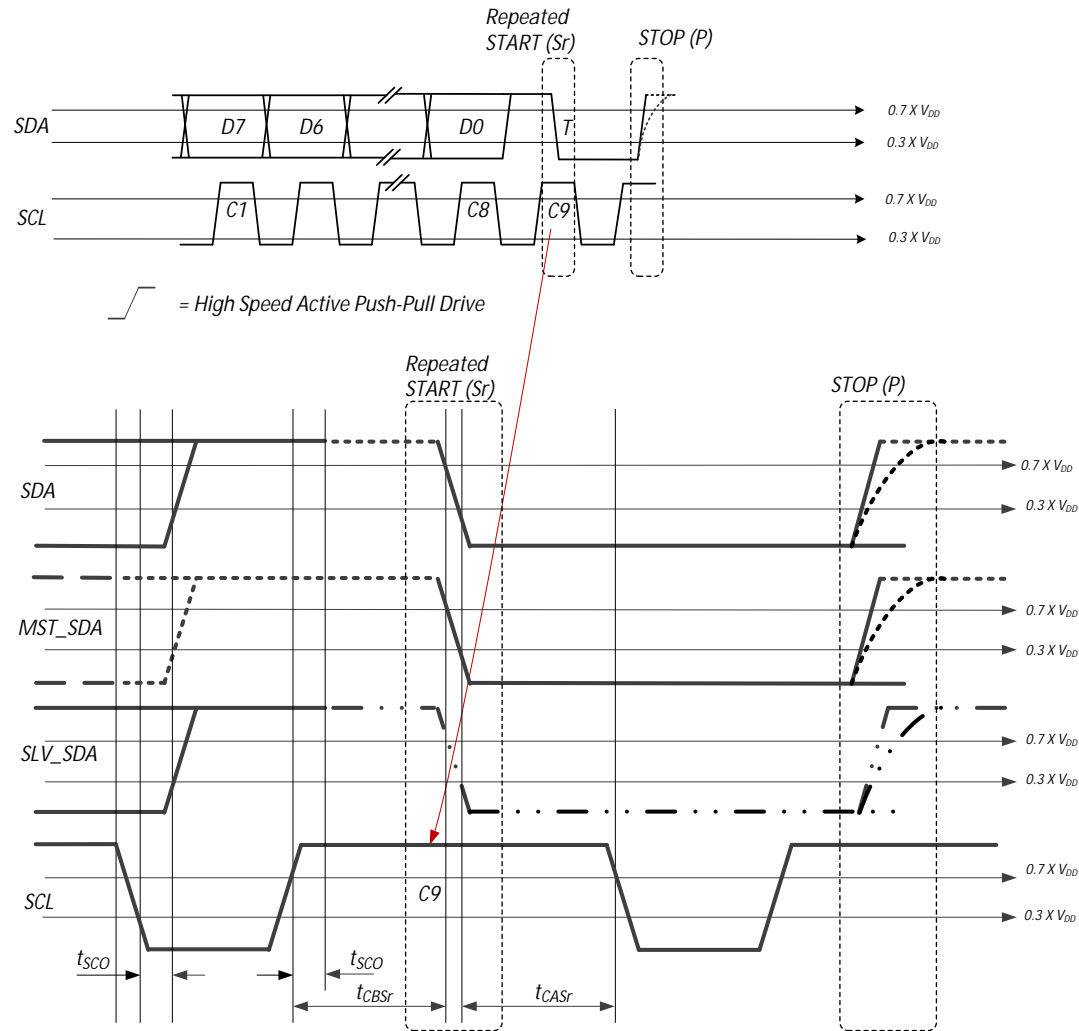


Slave terminates the Read transfer by pulling SDA Low in 9th bit and allowing Master Device to generate either STOP or RESTART

= Active Drive by Master or Slave
 = High-Z by Master
 = Open-Drain class Pull-Up by Master
 = High-Z by Slave

3114
3115

Figure 70 T-Bit When Slave Ends Read and Master Generates Repeated START

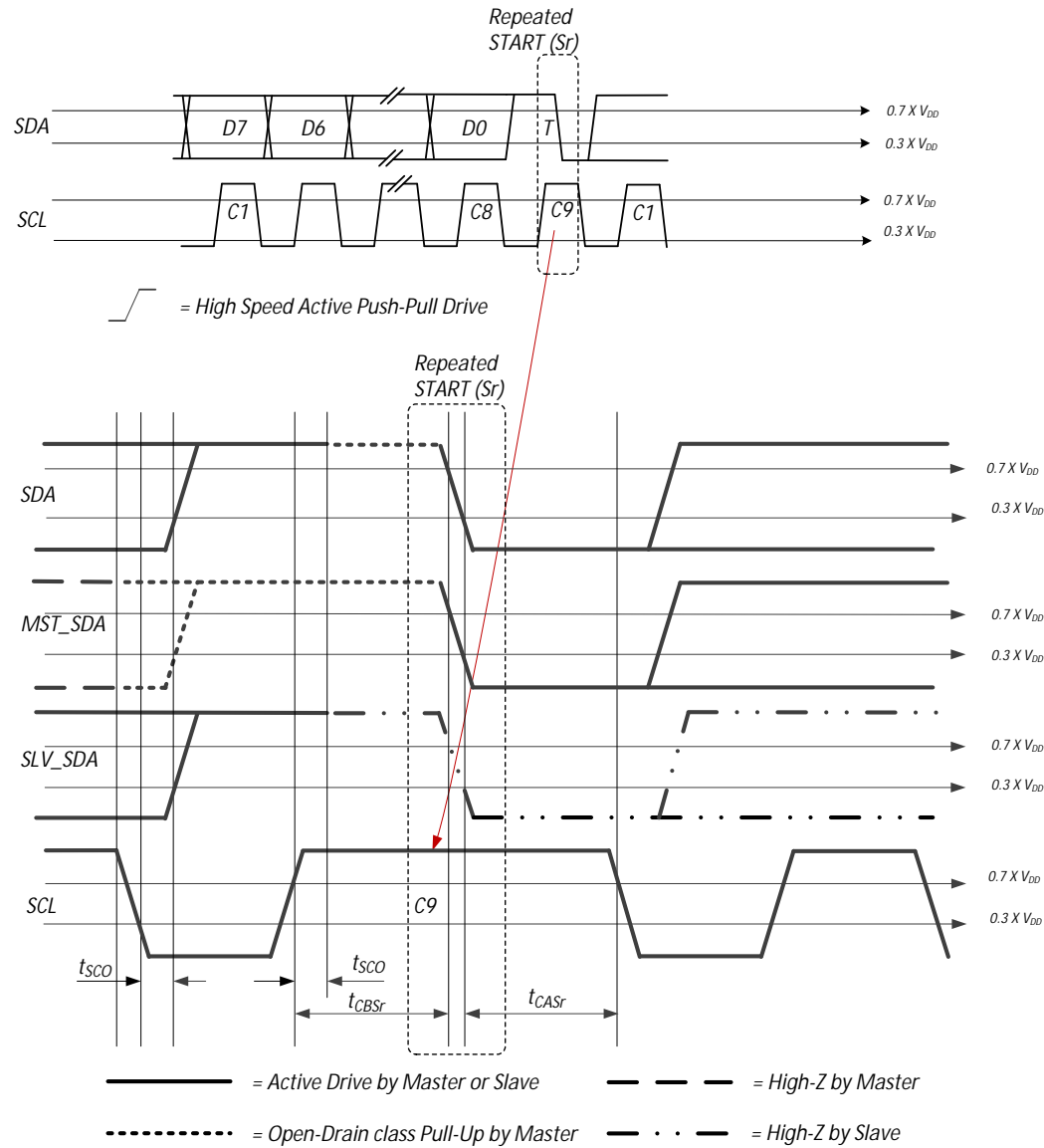


Note:
Sr immediately followed by P is illegal for I2C devices,
however many devices are not adversely affected by it.
The condition is not illegal for I3C devices

———— = Active Drive by Master or Slave
- - - - = High-Z by Master
- . . - = Open-Drain class Pull-Up by Master
- . . - = High-Z by Slave

3118
3119

Figure 72 T-Bit When Master Ends Read with Repeated START and STOP

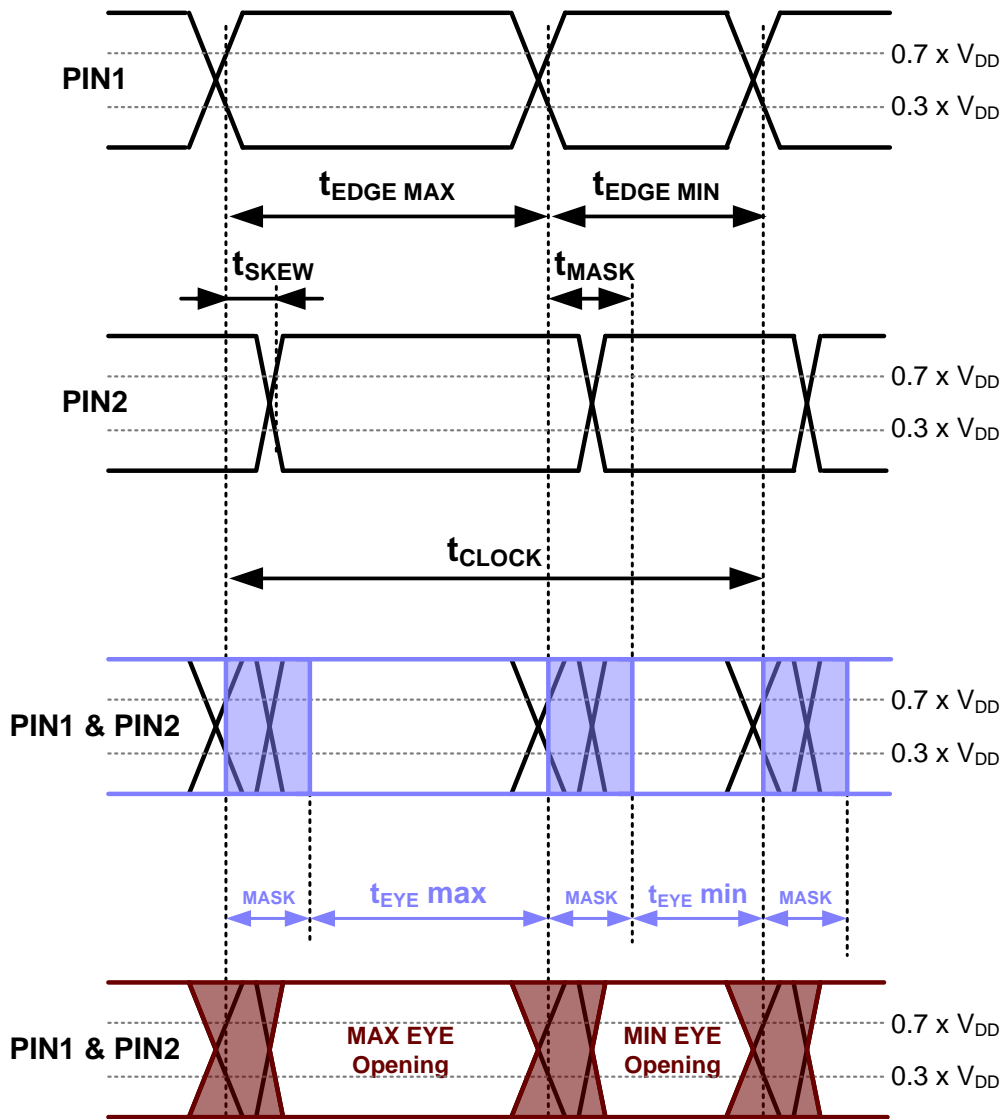


31.20
31.21

Figure 73 T-Bit When Master Ends Read via Repeated START and Further Transfer

3124
3125

Figure 75 shows the unique timing parameters for the I3C Ternary Protocol. In this Mode of operation, both the SDA line and the SCL line carry data.



Design Conditions

1. The differences of load magnitude and drive strength between the SDA and SCL lines result in different slopes, as shown
2. MASK period \geq MAX t_{SKEW} + MAX rise/fall time between $0.3 \times V_{DD}$ and $0.7 \times V_{DD}$, respectively, decreased by the time crossing the hysteresis window.

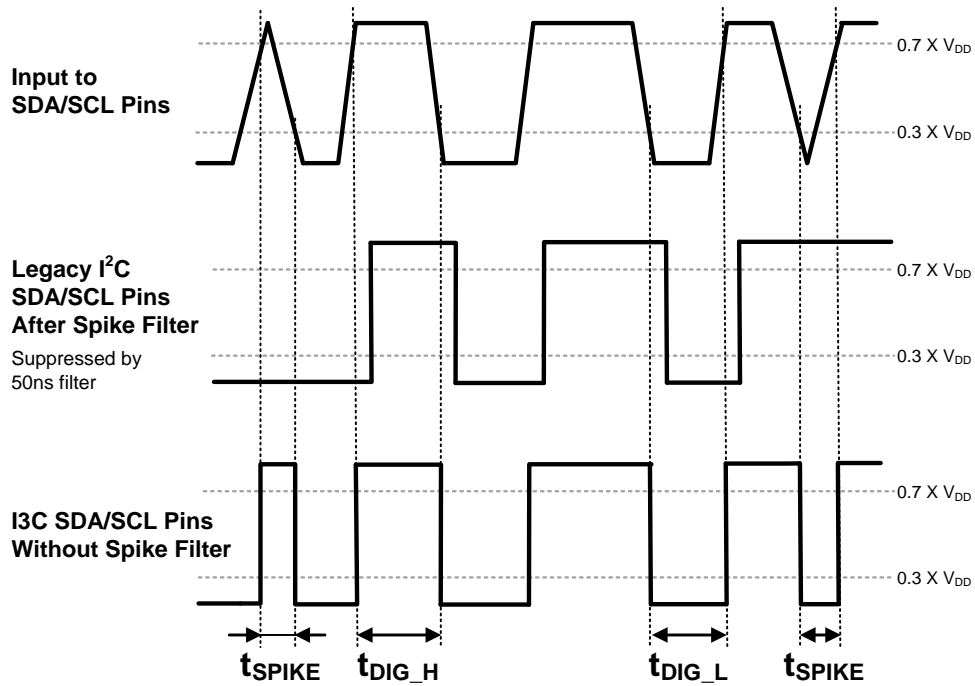
Simplifying Assumptions

1. t_{SKEW} is the same for both edges
2. Base CLOCK (e.g. 12.5MHz) is asymmetrical
3. Falling edges are equal to rising edges
4. The MASK is shown as starting from the last possible moment, whereas in reality it could be shifted left (towards the start point) delayed by the hysteresis window.

3126
3127

Figure 75 Ternary Protocol Timing

3128 **Figure 76** shows the timing parameters related to the Spike Filter on a Legacy I²C Device. This timing shall
3129 be met, in order to ensure that Legacy I²C Devices properly ignore I3C High Speed Mode. It is recommended
3130 that both the SCL pin and the SDA pin have a Spike Filter. The timing is specified in **Table 75**.



3131
3132

Figure 76 I²C Spike Filter Behavior

3133 **Table 77** through **Table 79** detail how timing and drive strength are adjusted during transmission of an I3C
 3134 Message.

3135 **Table 77 Timing and Drive for Start of New Frame: No Contention on A7**

	S	Header				Data	P
	START	ArbBit	Addr [5:0]	RnW	ACK	N Data + Parity	STOP
SDA Mode	Open Drain	Open Drain	Push-Pull	Push-Pull	Open Drain	Push-Pull	Push-Pull
Clocks	1	1	6	1	1	9 * N	1

3136 **Table 78 Timing and Drive for Start of New Frame: With Contention on A7**

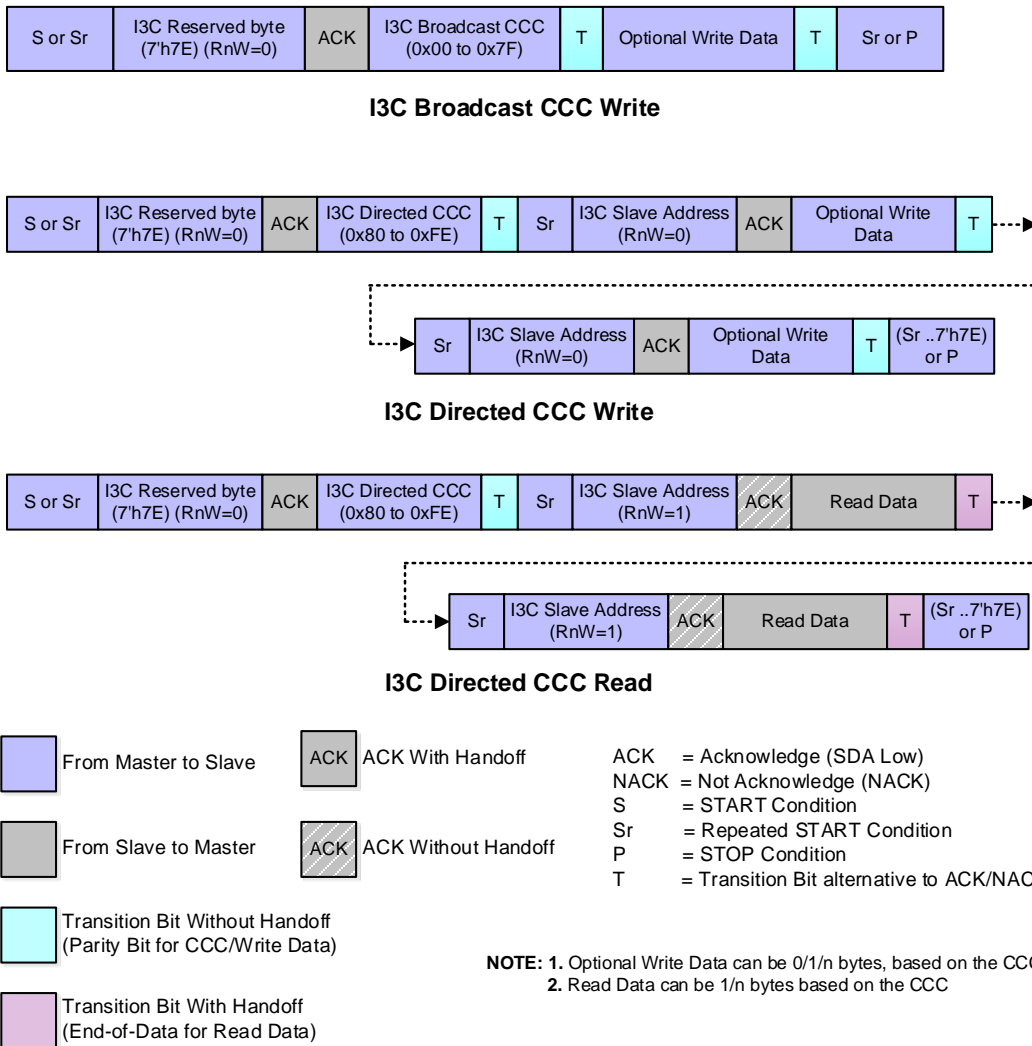
	S	Header				Data	P
	START	ArbBit	Addr [5:0]	RnW	ACK	N Data + Parity	STOP
SDA Mode	Open Drain	Open Drain	Open Drain	Open Drain	Open Drain	Push-Pull	Push-Pull
Clocks	1	1	6	1	1	9 * N	1
Note: <i>Contention on A6, so Arbitrated over A5 to A0 and RnW.</i>							

3137 **Table 79 Timing and Drive for Continuation of Frame Using Repeated START**

	S	Header / Data...	Sr	Header		Data	P
	START	...	START	Addr/RW	ACK	N Data + Parity	STOP
SDA Mode	Open Drain	...	Push-Pull	Push-Pull	Open Drain	Push-Pull	Push-Pull
Clocks	1	...	1	8	1	9 * N	1
Note: <i>There may be any number of Repeated STARTs before the STOP.</i>							

Annex A I3C Communication Format Details

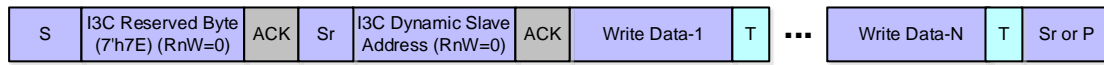
A.1 I3C CCC Transfers



3138
3139

Figure 77 I3C CCC Transfers

A.2 I3C Private Write and Read Transfers



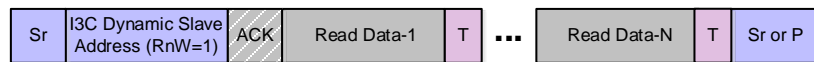
I3C Private Write Transfer Initiated with START Condition



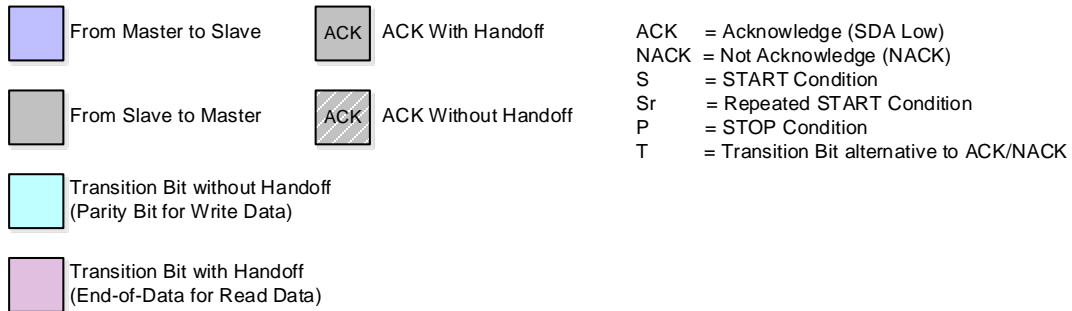
I3C Private Write Transfer Initiated with Repeated START Condition



I3C Private Read Transfer Initiated with START Condition



I3C Private Read Transfer Initiated with Repeated START Condition



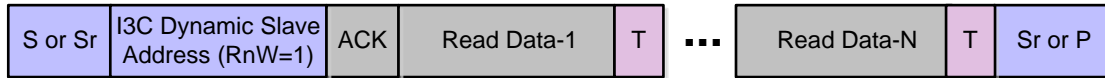
3140

3141

Figure 78 I3C Private Write and Read Transfers with 7'h7E Address



I3C Private Write Transfer



I3C Private Read Transfer Initiated with Repeated START Condition



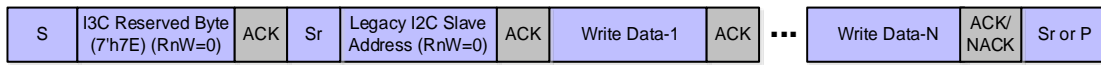
ACK = Acknowledge (SDA Low)
 NACK = Not Acknowledge (NACK)
 S = START Condition
 Sr = Repeated START Condition
 P = STOP Condition
 T = Transition Bit alternative to ACK/NACK

3142

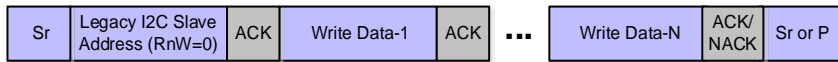
3143

Figure 79 I3C Private Write and Read Transfers without 7'h7E Address

A.3 Legacy I²C Write and Read Transfers on the I3C Bus



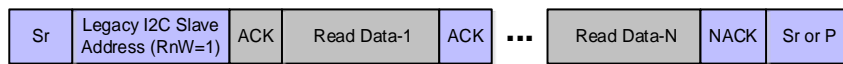
Legacy I2C Write Transfer Initiated with START Condition



Legacy I2C Write Transfer Initiated with Repeated START Condition



Legacy I2C Read Transfer Initiated with START Condition

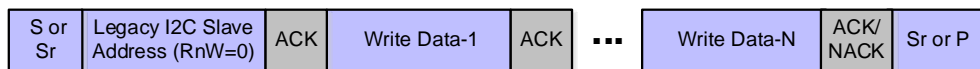


Legacy I2C Read Transfer Initiated with Repeated START Condition

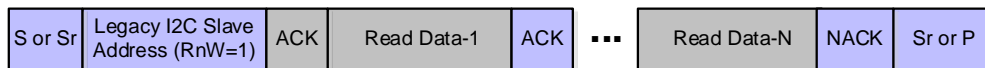


3144
3145

Figure 80 Legacy I²C Write and Read Transfers in I3C Bus with 7'h7E Address



Legacy I2C Write Transfer



Legacy I2C Read Transfer



3146
3147

Figure 81 Legacy I²C Write and Read Transfers in I3C Bus without 7'h7E Address

A.4 Dynamic Address and Enter HDR

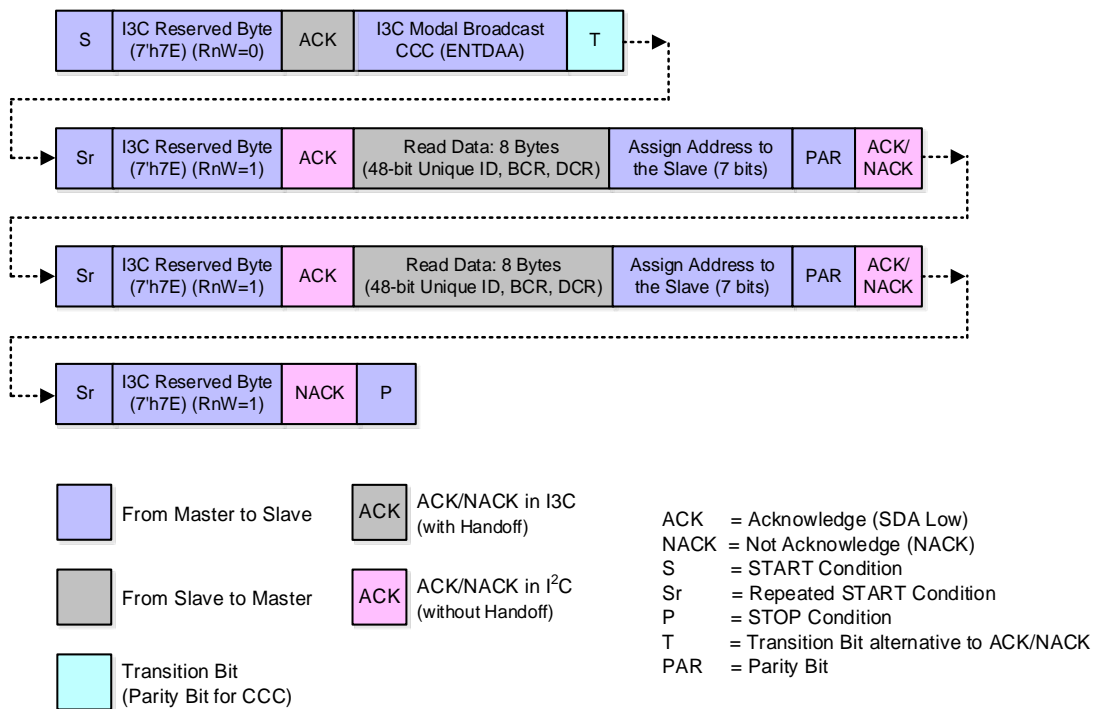


Figure 82 Dynamic Address Allocation ENTDAACCC Bus Modal

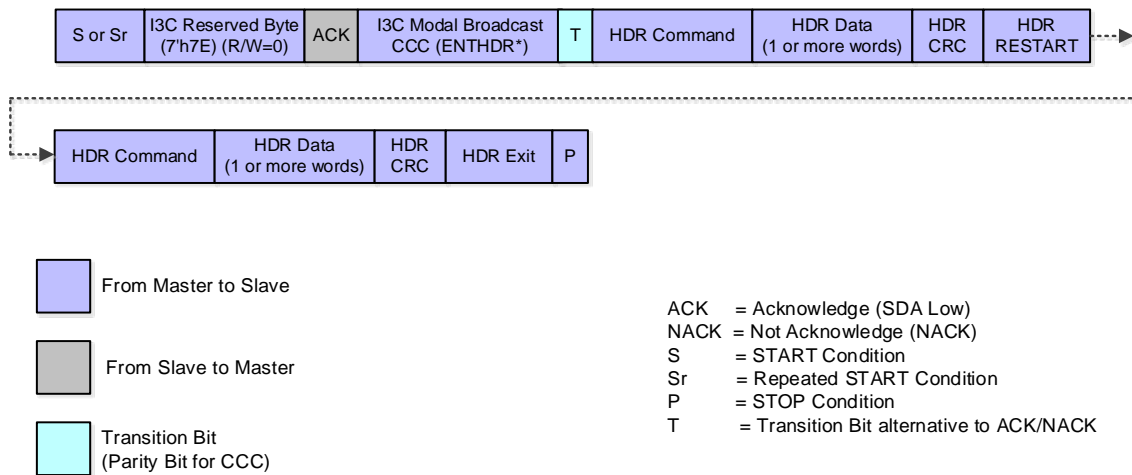
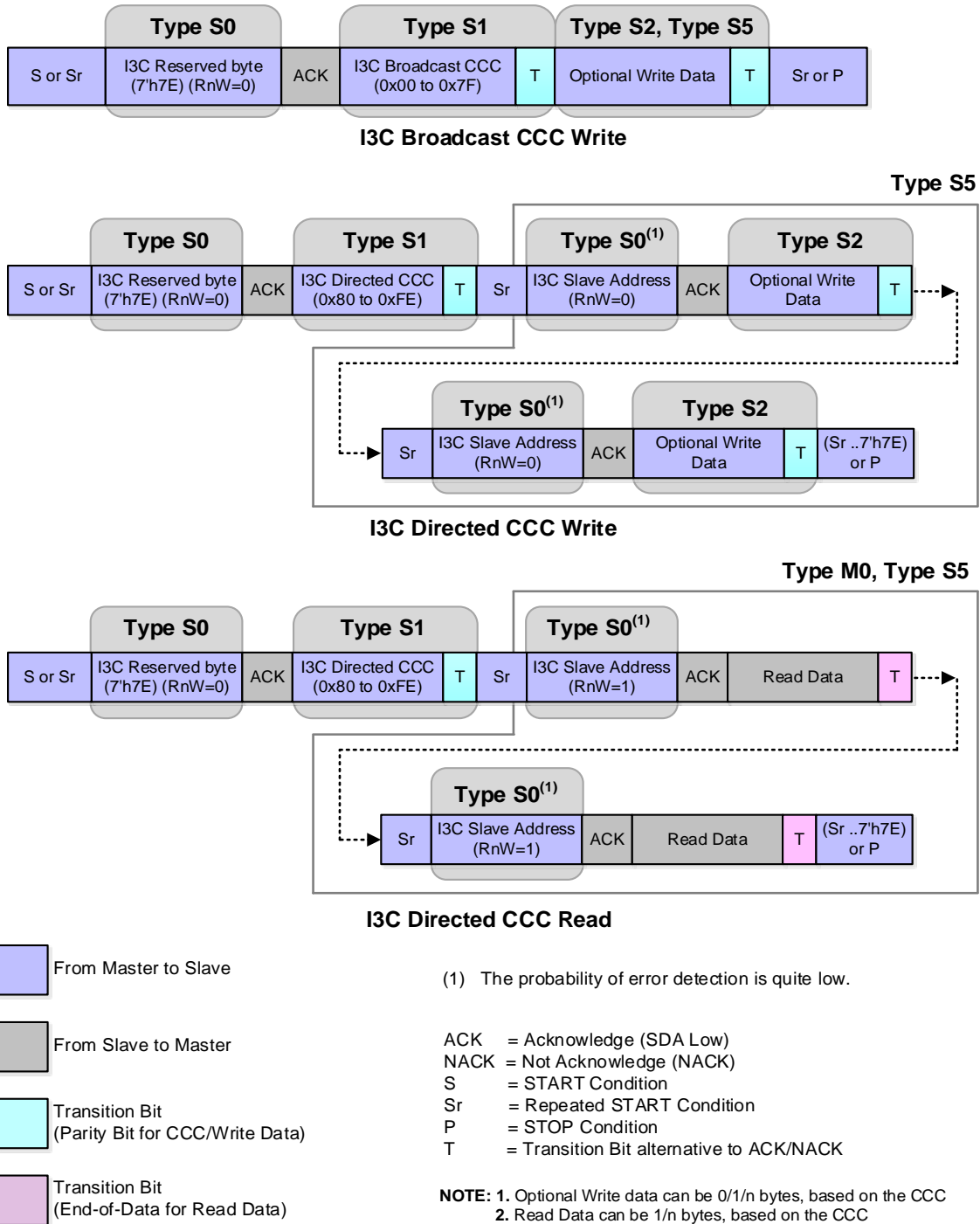


Figure 83 Enter HDR Mode CCC Bus Modal

This page intentionally left blank.

Annex B SDR Mode Error Type Origins

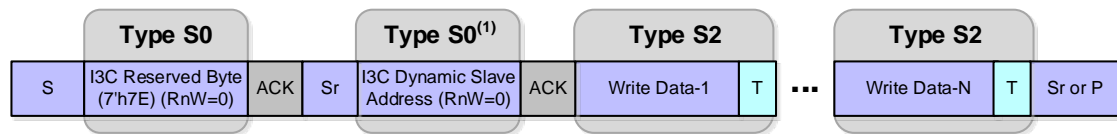
B.1 Error Types in I3C CCC Transfers



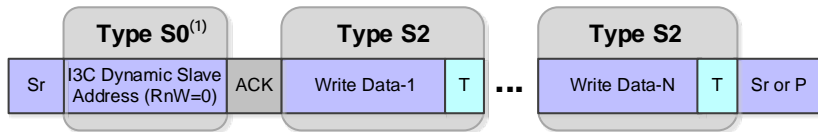
3152
3153

Figure 84 Error Type Origins: I3C CCC Transfers

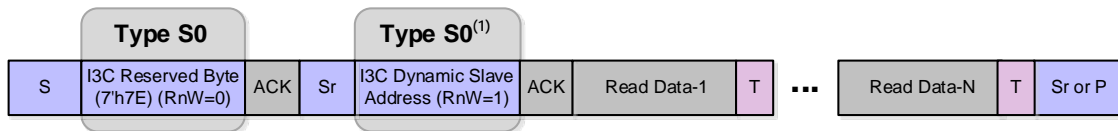
B.2 Error Types in I3C Private Read and Write Transfers



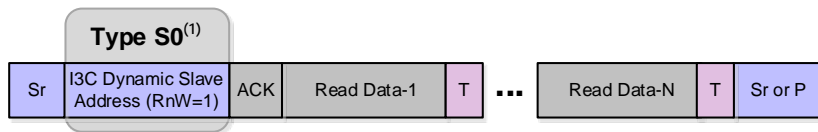
I3C Private Write Transfer Initiated with START Condition



I3C Private Write Transfer Initiated with Repeated START Condition



I3C Private Read Transfer Initiated with START Condition



I3C Private Read Transfer Initiated with Repeated START Condition



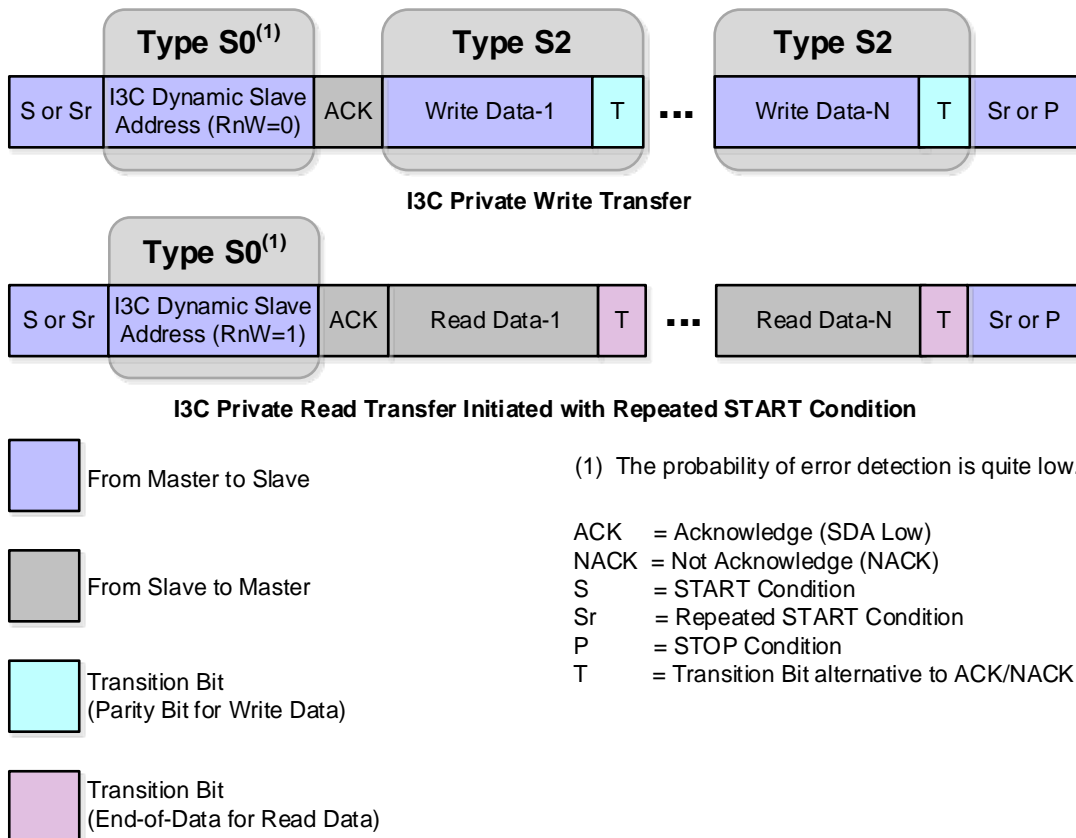
(1) The probability of error detection is quite low.

ACK = Acknowledge (SDA Low)
 NACK = Not Acknowledge (NACK)
 S = START Condition
 Sr = Repeated START Condition
 P = STOP Condition
 T = Transition Bit alternative to ACK/NACK

3154

3155

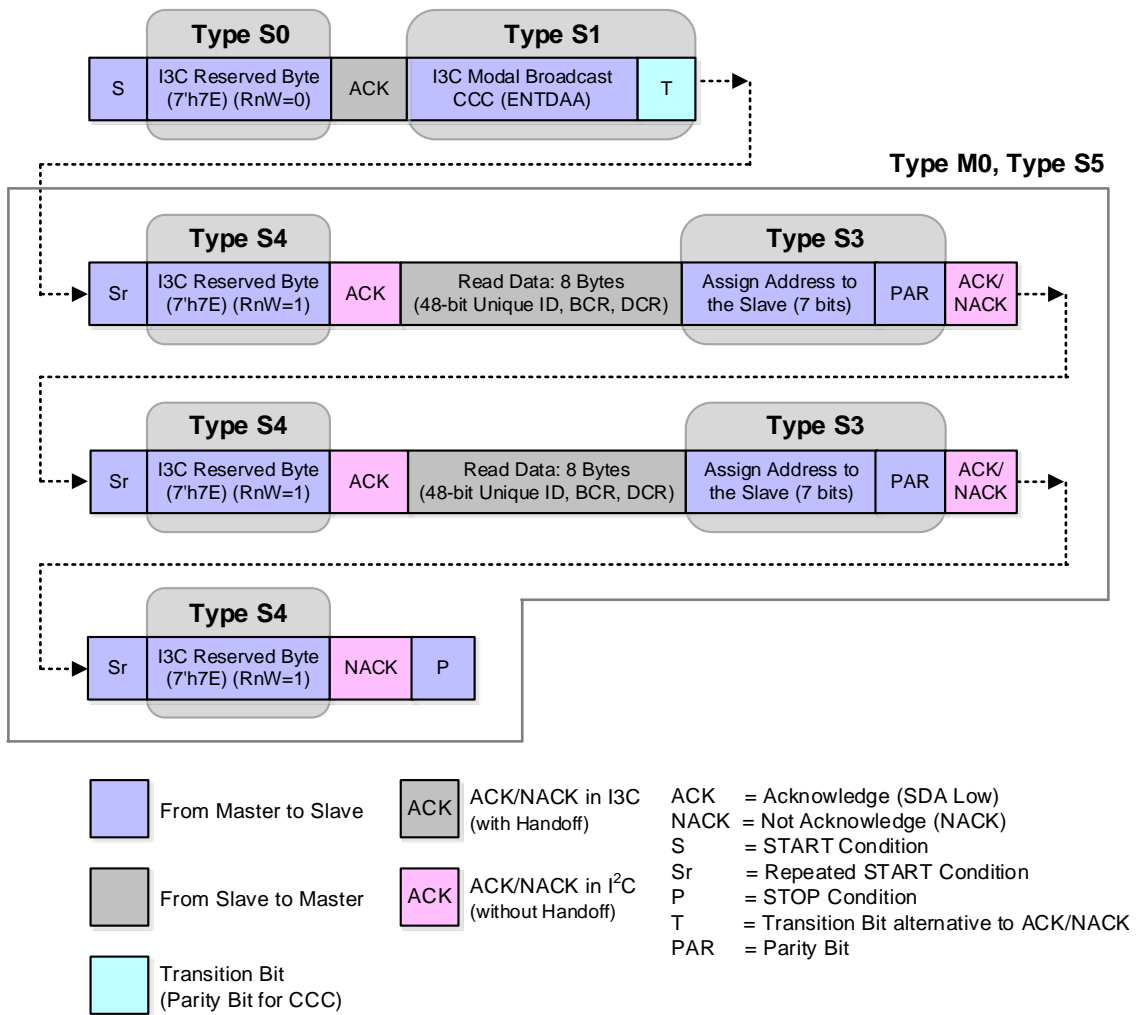
Figure 85 Error Type Origins: I3C CCC Private Write & Read Transfers with 7'h7E Address



3156
3157

Figure 86 Error Type Origins: I3C Private Read Transfers without 7'h7E Address

B.3 Error Types in Dynamic Address Arbitration



3158

3159

Figure 87 Error Type Origins: Dynamic Address Arbitration

Annex C I3C Master FSMs

3160 **Figure 88** shows the key transmission Modes and their entry, resume and exit sequences. It includes the SDR
 3161 transmission states, and differentiates these from other capabilities and Modes. It captures the states at which
 3162 Arbitration happens as well.

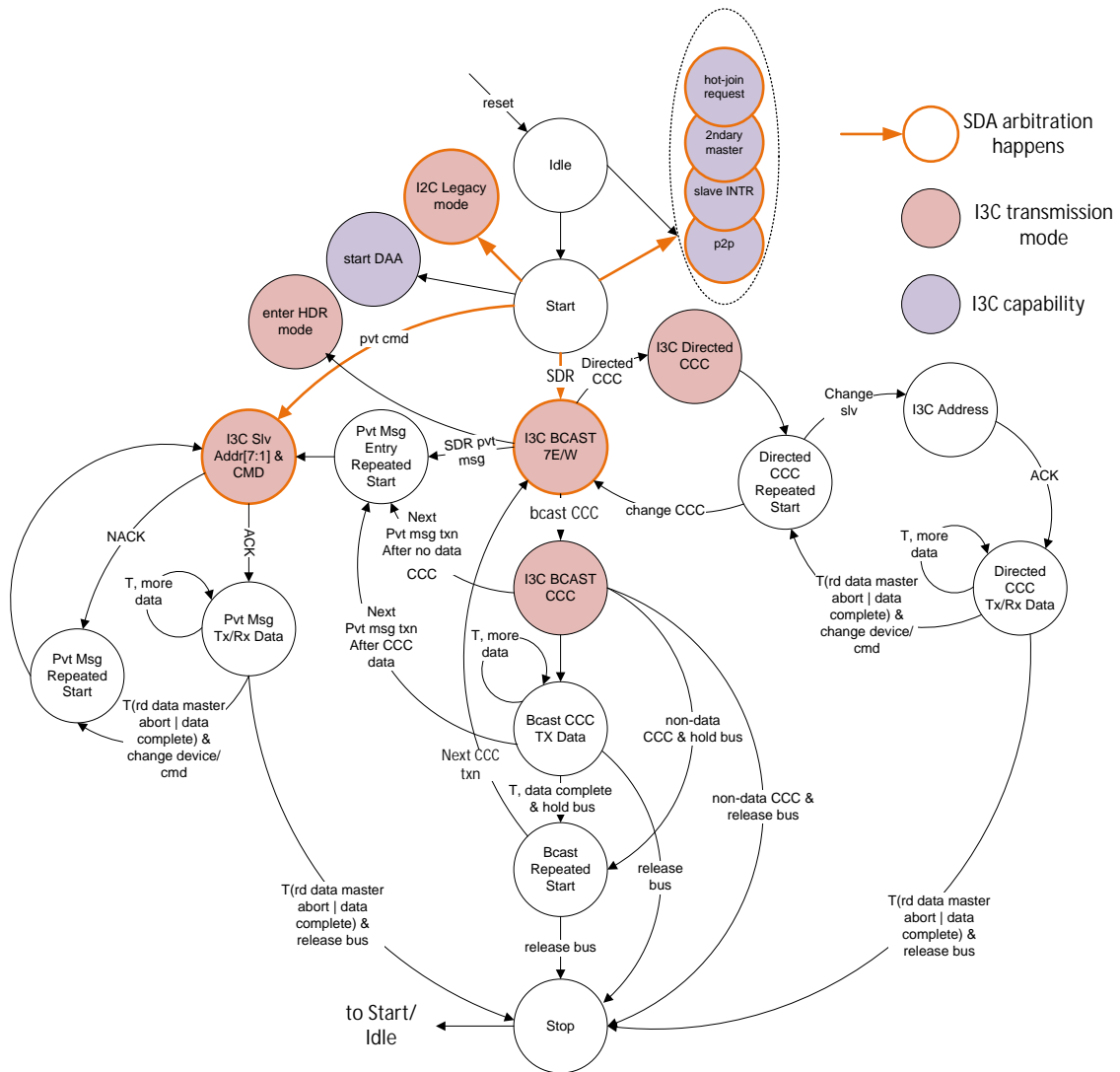


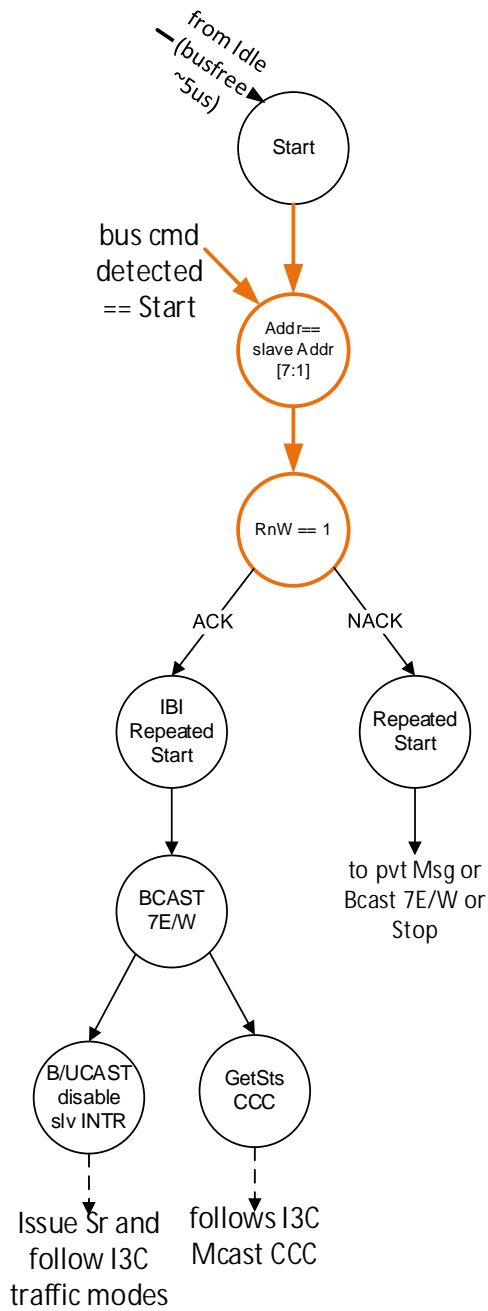
Figure 88 I3C Main Master FSM

3163
3164

3165 **Figure 89** through **Figure 93** represent I3C features including In-Band Interrupts, Secondary Master,
 3166 Dynamic Address Assignment, and Hot-Join.

3167

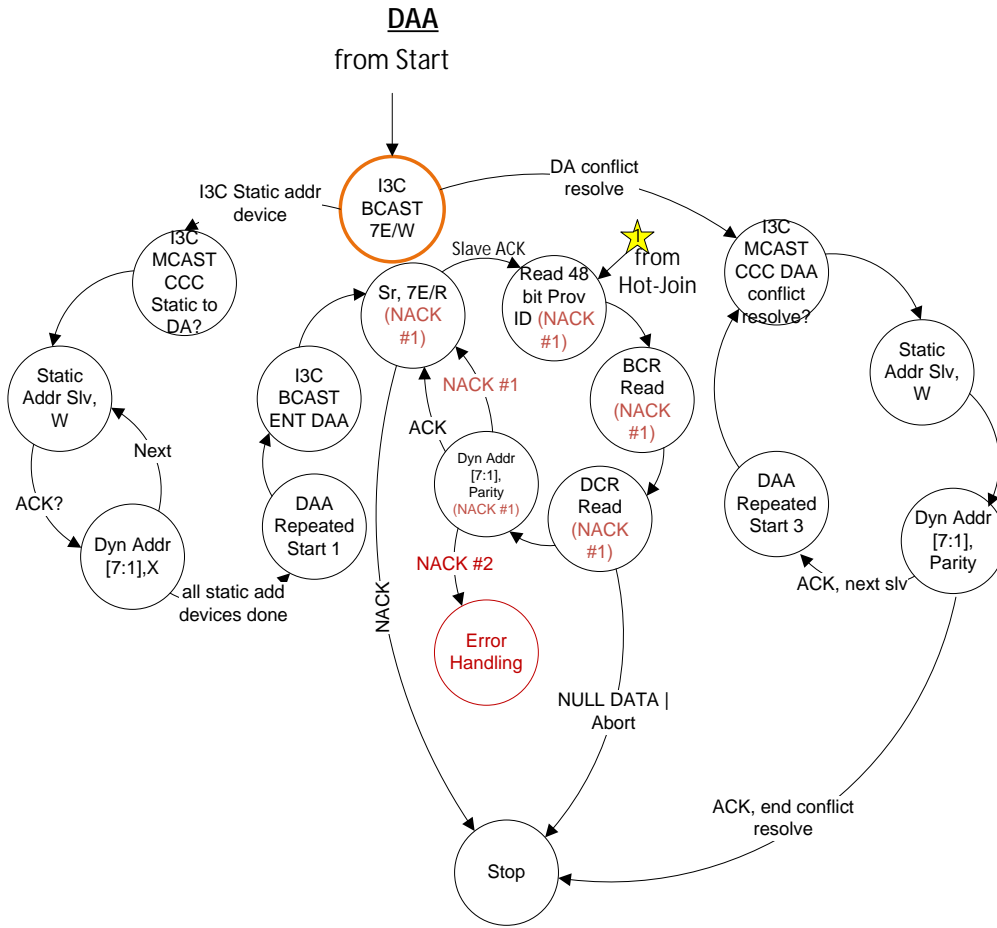
Slave Interrupt request



3168

3169

Figure 89 Slave Interrupt Request FSM



3170
3171

Figure 90 Dynamic Address Assignment FSM

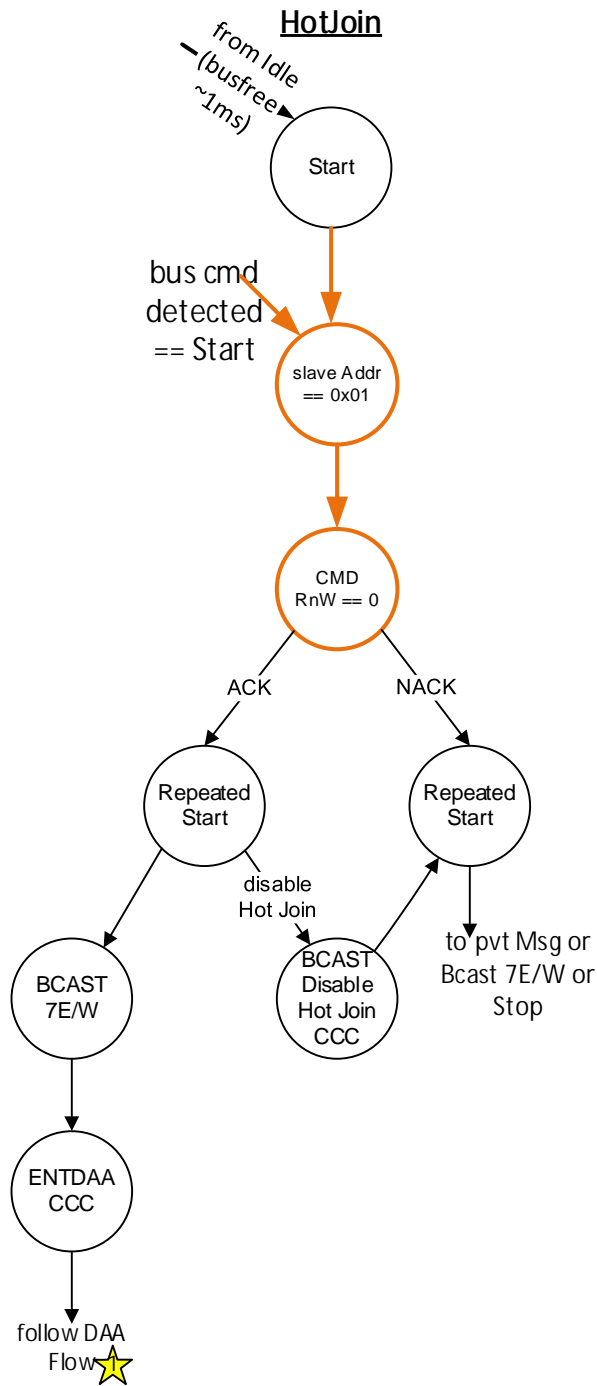
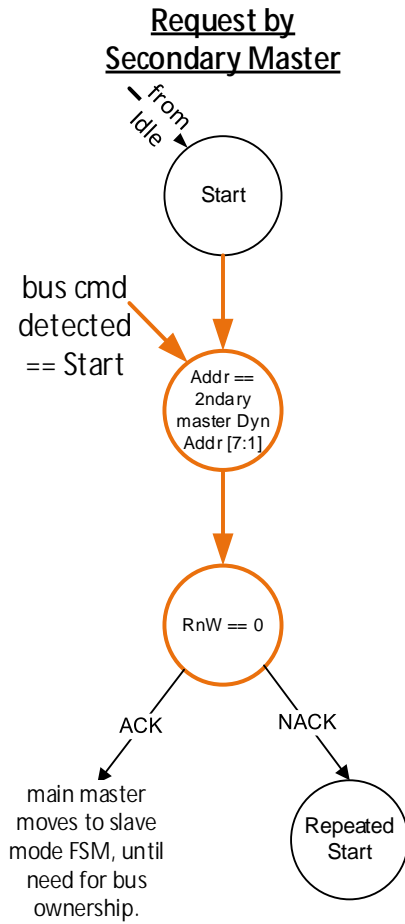


Figure 91 Hot-Join FSM

31.72
31.73



3174
3175

Figure 92 Secondary Master Request FSM

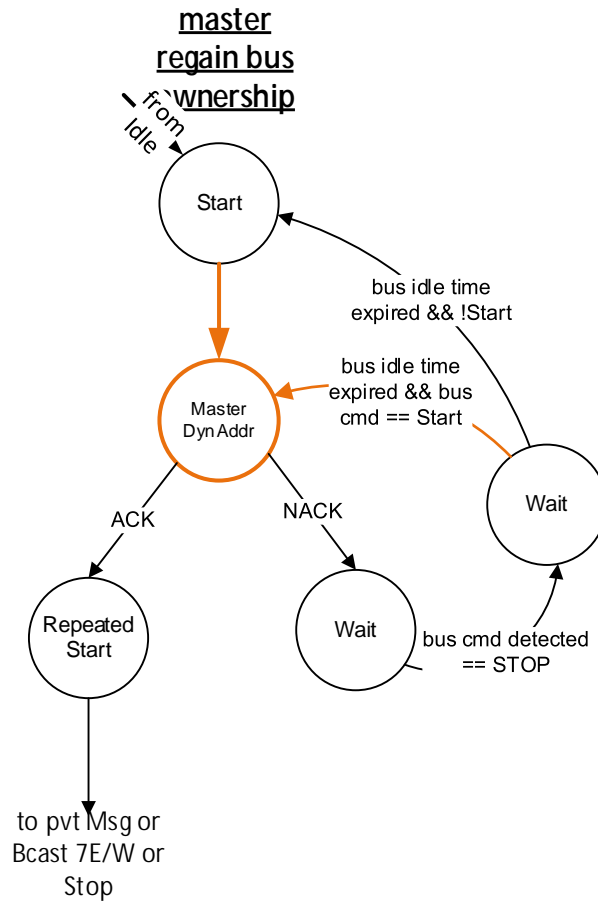
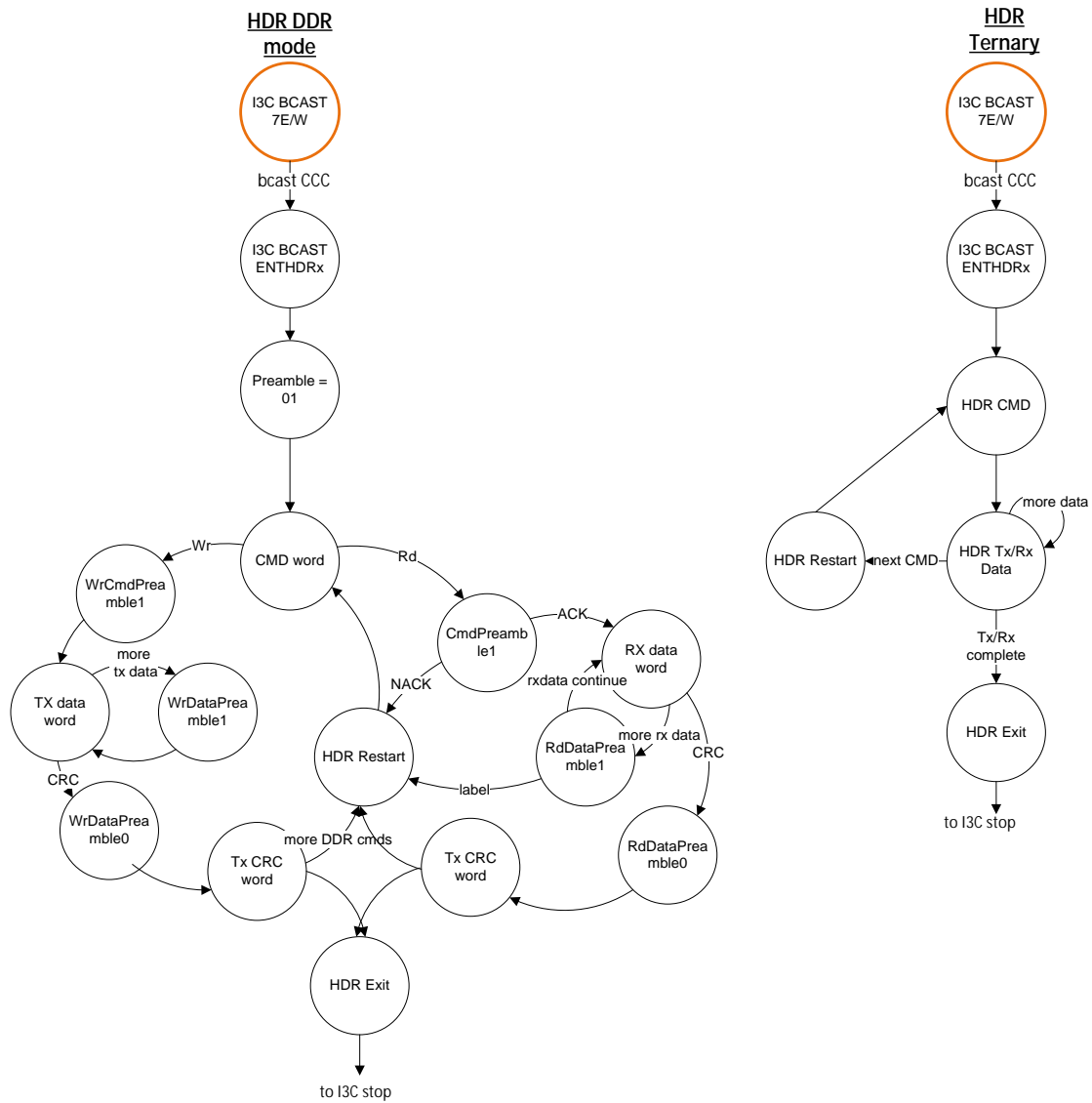


Figure 93 Master Regaining Bus Ownership FSM

3176
3177

3178 The FSMs in **Figure 94** represent the Master states when in HDR transmission Modes, including Entry,
3179 Resume, and Exit sequence for HDR Ternary Modes and HDR-DDR Mode.

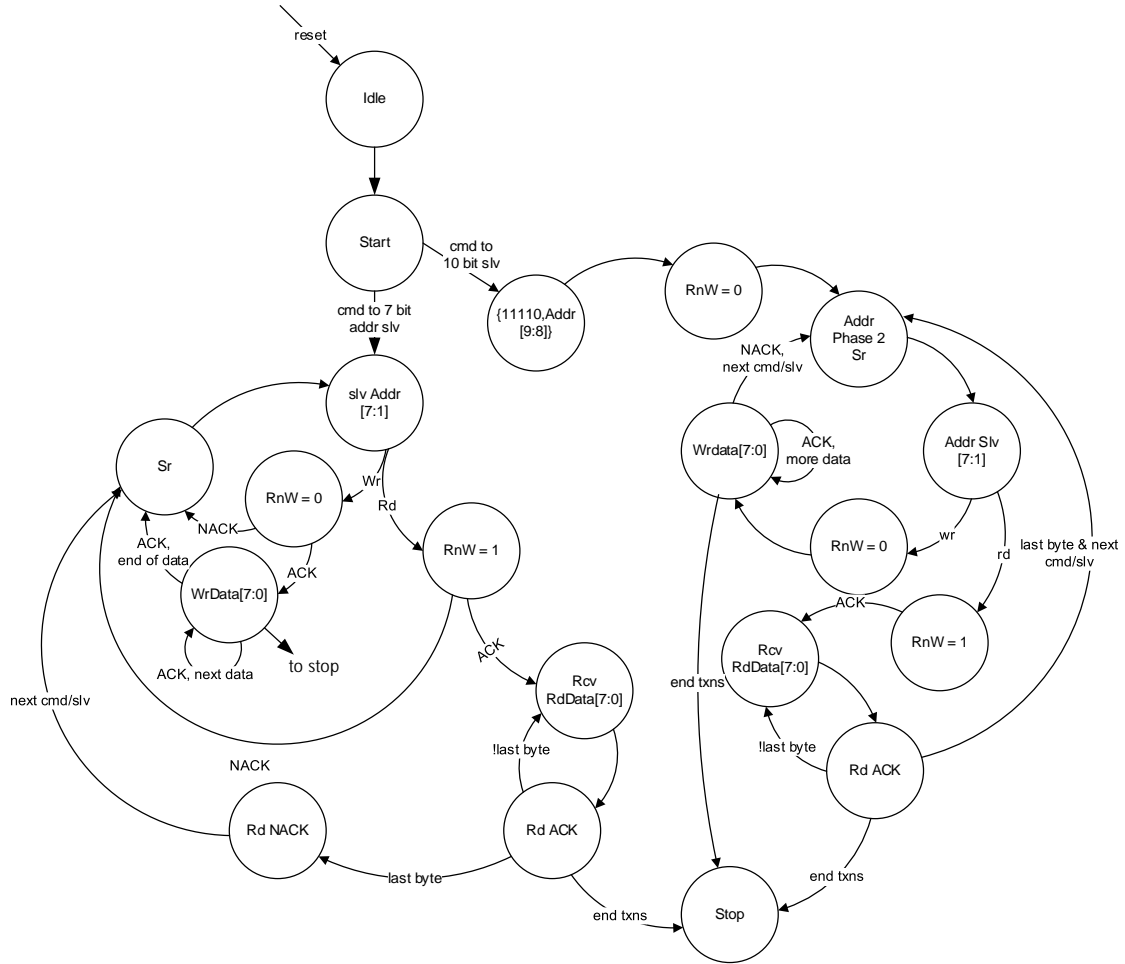


3180
3181

Figure 94 HDR Ternary and HDR-DDR Mode FSMs

3182 **Figure 95** is for reference only.

I2C legacy Master FSM



3183

3184

Figure 95 I2C Legacy Master FSM

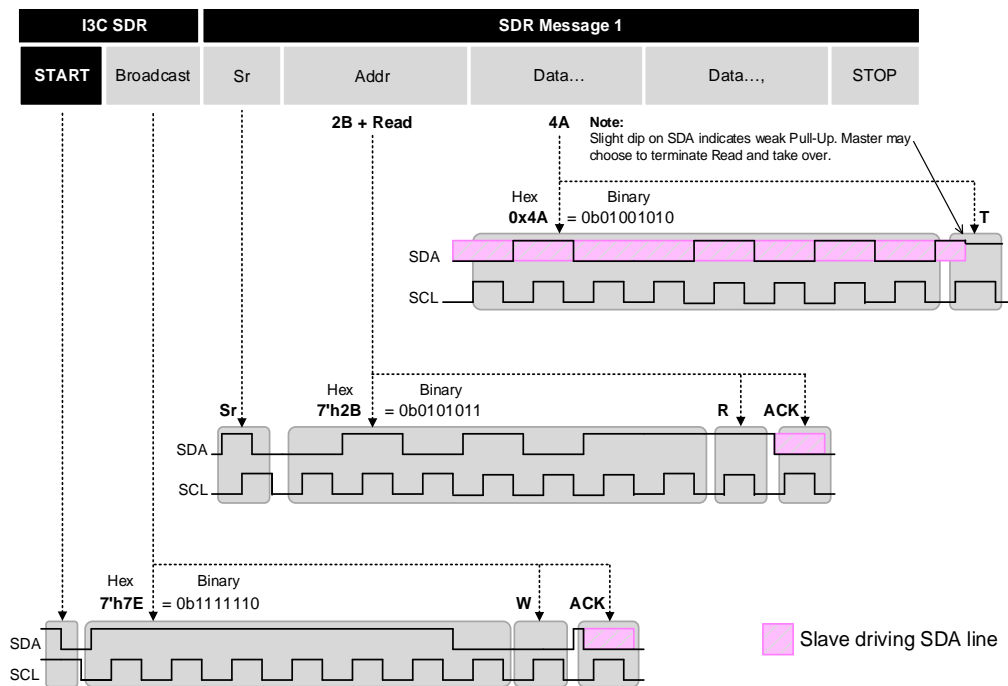
Annex D Typical I3C Protocol Communications

3185 **Figure 96** through **Figure 101** illustrate a typical communication for each of the six I3C Protocols. While
3186 these diagrams do not exhaustively illustrate all possible I3C communications, they do serve as useful
3187 introductions to the signaling and transmission formatting used in each I3C Protocol.

3188 **Figure 96** illustrates example communication using I3C Single Data Rate (SDR) coding. It shows the Master
3189 reading a byte of data from the Slave at Address 0x2B in SDR Mode.

3190 From the Bus Free Condition, the Master issues a START by driving the SDA line Low while keeping the
3191 SCL line High. It then issues the Broadcast Address (7'h7E) followed by RnW (0 for Write). Then the Master
3192 turns on a pull-up resistor and goes to Open Drain. All Slaves ACK by pulling the SDA line Low (in the
3193 Figure, pink fill means the Slave is in control of the SDA line at this time). The Master then issues a Repeated
3194 START, then the Address of the Slave (0x2B) it wants to read followed by RnW (1 for Read). The Master
3195 then turns on a pull-up resistor and goes to Open Drain, allowing the Slave to acknowledge by pulling the
3196 SDA line Low. At this point, the Master continues to toggle the SCL line and release the SDA line, allowing
3197 the Slave to drive SDA to send one byte of data (0x4A) followed by 'T'. T=1 informs the Master that there
3198 is additional data, whereas T=0 signals the end. Here there is additional data, so the Slave drives SDA High
3199 until SCL goes High, at which time it releases SDA. The Master has the option of holding SDA High with a
3200 weak pull-up, which signals to the Slave that the Master allows another byte to be transmitted, or to pull SDA
3201 Low (while SCL is High – hence a Repeated START), which would signal to the Slave that the Master has
3202 terminated the Read and is taking over.

3203 SDR Mode is backwards compatible with Legacy I²C Devices, because the High time of an SCL pulse is
3204 always less than 50ns and therefore SCL will always appear to be Low because of the I²C 50ns Spike Filter.

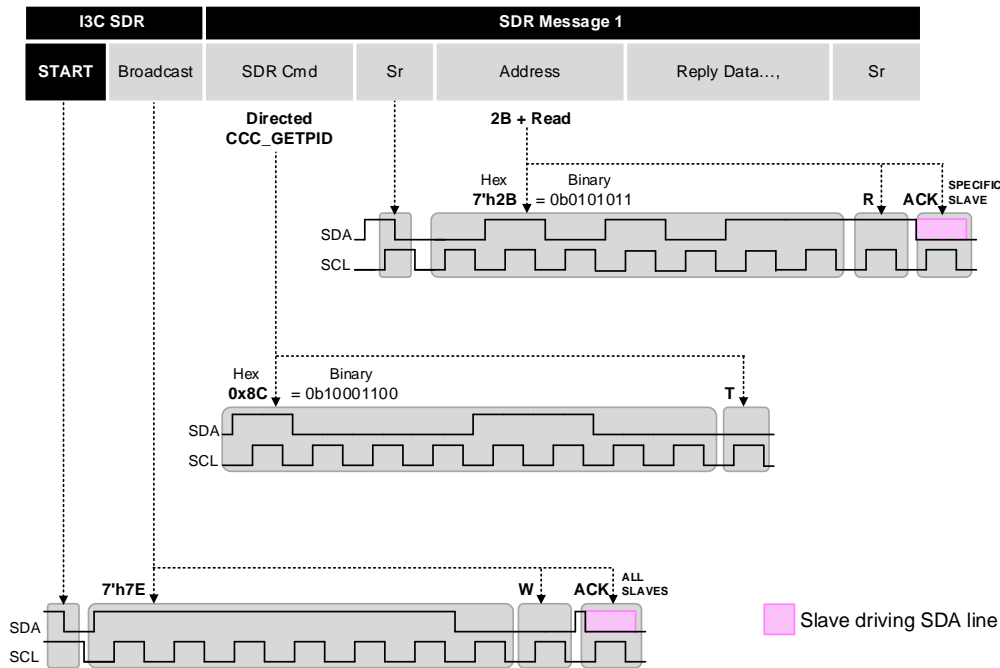


3205
3206

Figure 96 Example Communication Using I3C Coding SDR

3207 **Figure 97** shows the Master issuing a CCC Direct Command to a single Slave. This particular command
3208 (GETPID) reads the Provisional ID of a Slave.

3209 From the Bus Free Condition, the Master issues a START by driving the SDA line Low while keeping the
3210 SCL line High. It then issues the Broadcast Address (7'h7E) followed by RnW (0 for Write). Then the Master
3211 turns on a pull-up resistor and goes to Open Drain. All Slaves ACK by pulling SDA Low (in the Figure, pink
3212 fill means the Slaves are in control of SDA at this time). The Master then issues the Direct Common
3213 Command Code for GETPID (0x8C) followed by parity bit 'T' (odd parity = 0 for 0x8C) then the 7-bit
3214 Dynamic Address of the Slave (chosen arbitrarily here to be 0x2B) followed by a RnW bit (1 for Read). Then
3215 the Master turns on a pull-up resistor and goes to Open Drain, allowing the Slave at Address 0x2B to ACK
3216 by pulling SDA Low, which tells the Master that the Slave Acknowledges the command and will comply.
3217 (Alternatively, the Slave may NACK by not pulling SDA Low, which would inform the Master that the Slave
3218 will not comply – in this case, that an error occurred.) Following the ACK the Slave outputs its 48-bit PID
3219 one byte at a time, and then the Master issues a Repeated START (this part of the waveform sequence is not
3220 shown in the Figure).

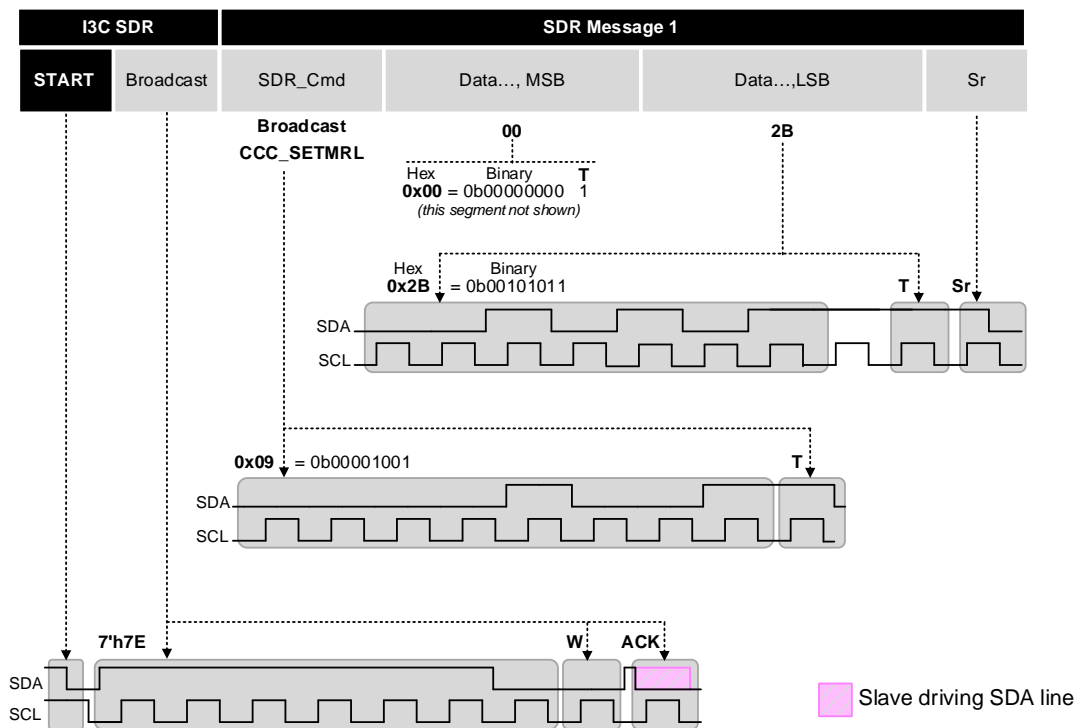


3221
3222

Figure 97 Example Communication Using I3C Coding SDR with CCC Direct Addressing

3223 **Figure 98** illustrates example SDR communication with a CCC Broadcast command. The command used in
3224 this example sets the Maximum Read Length of all Slaves to 43 bytes (0x002B).

3225 From the Bus Free Condition, the Master issues a START by driving the SDA line Low while keeping the
3226 SCL line High. It then issues the Broadcast Address (7'h7E) followed by RnW (0 for Write). Then the Master
3227 turns on a pull-up resistor and goes to Open Drain. All Slaves ACK by pulling SDA Low (in the Figure, pink
3228 fill means the Slaves are in control of SDA at this time). The Master then issues the Broadcast Common
3229 Command Code for SETMRL (0x09) followed by parity bit 'T' (odd parity = 1 for 0x09), and then 2 data
3230 bytes (MSb first) to define the maximum number of bytes which can be read from a Slave in a single read
3231 operation. Each data byte is followed by a 'T' bit (parity bit – odd parity). After this the Master issues a
3232 Repeated START.



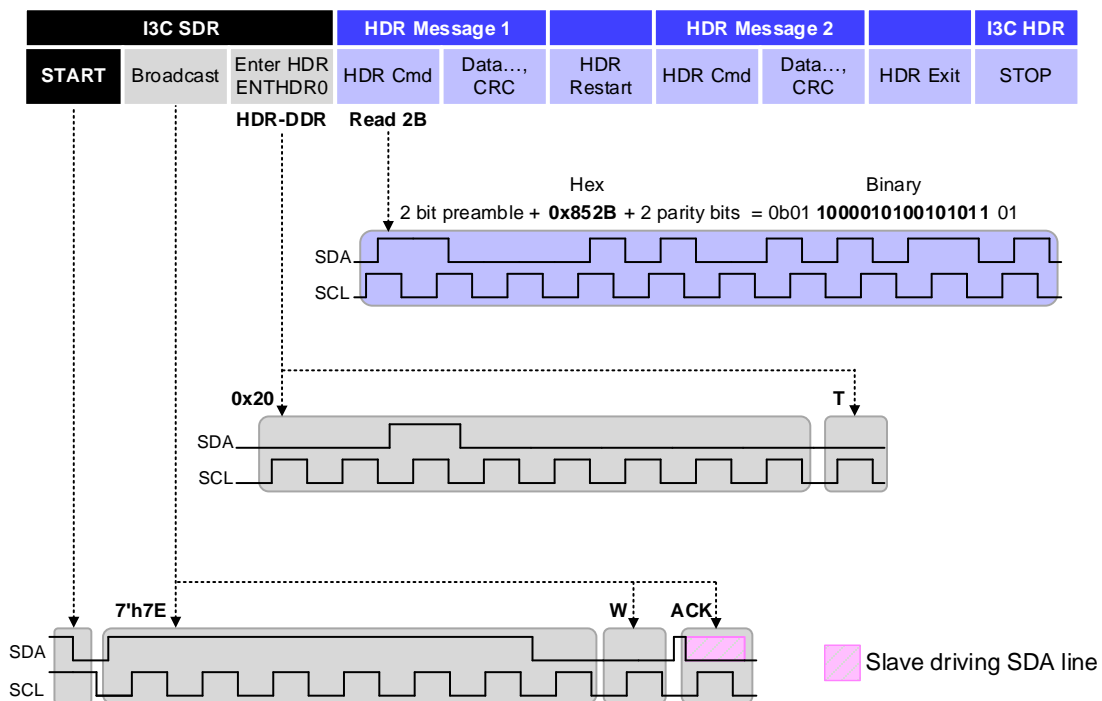
3233
3234

Figure 98 Example Communication Using I3C Coding SDR with CCC Broadcast

3235 **Figure 99** illustrates use of the HDR-DDR (Double Data Rate) Mode. It shows how the Master can change
 3236 the Mode from SDR (Single Data Rate) Mode to HDR-DDR Mode, and a sample of the HDR-DDR data
 3237 format.

3238 From the Bus Free Condition, the Master issues a START by driving the SDA line Low while keeping the
 3239 SCL line High. The Master then issues the Broadcast Address (7'h7E) followed by RnW (0 for Write). Then
 3240 the Master turns on a pull-up resistor and goes to Open Drain. All Slaves 'ACK' by pulling SDA Low (in the
 3241 Figure, pink fill means the Slaves are in control of SDA at this time). The Master then issues the Broadcast
 3242 Common Command Code for ENTHDR0 (0x20), followed by parity bit "T" (odd parity = 0 for 0x20). At
 3243 this point, the Bus is in HDR-DDR Mode. In the HDR-DDR protocol, the SDA line is sampled on every SCL
 3244 edge (both Low-to-High and High-to-Low transitions of SCL). The HDR-DDR Word consists of a 2-bit
 3245 Preamble, followed by two bytes of data, followed by two parity bits. The waveform for the 5-bit CRC and
 3246 following traffic is not shown in the Figure.

3247 HDR-DDR Mode is backwards compatible with Legacy I²C Devices, because the High time of an SCL pulse
 3248 is always less than 50ns and as a result SCL will always appear to be Low because of the I²C 50ns Spike
 3249 Filter.



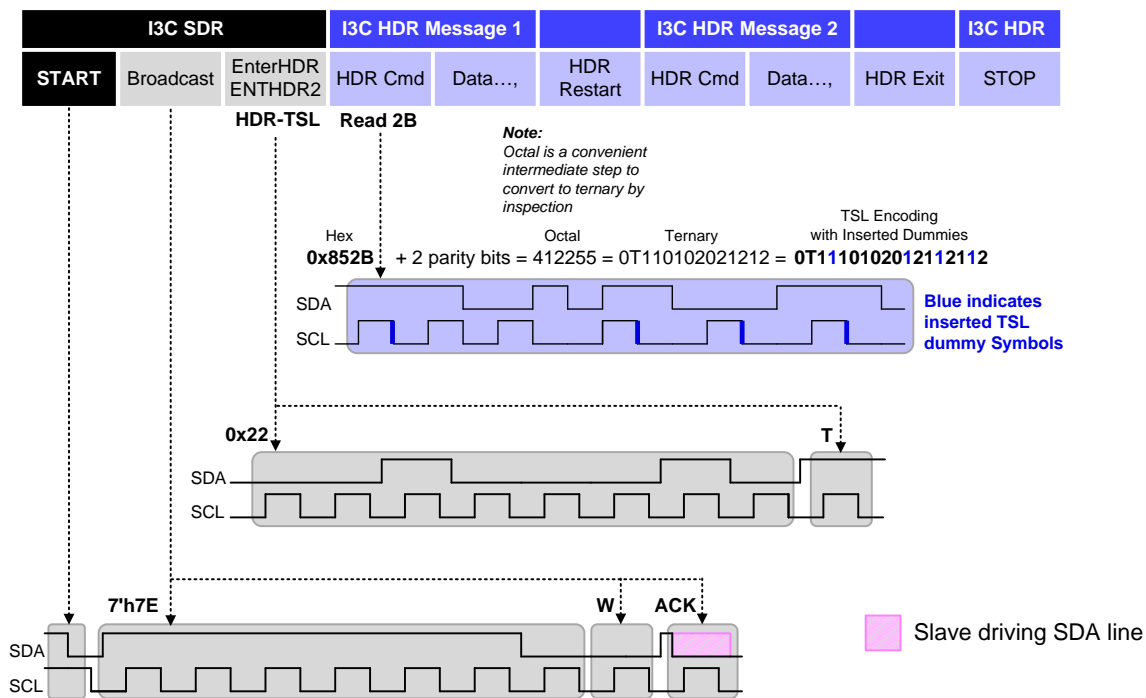
3250
3251

Figure 99 Example Communication Using HDR-DDR Protocol

3252 **Figure 100** illustrates use of the HDR-TSL (Ternary Symbol Legacy) Mode, which is only used if a Legacy
 3253 I²C Device is present on the Bus. It shows how the Master can change the Mode from SDR (Single Data
 3254 Rate) Mode to HDR-TSL Mode, and a sample of the HDR-TSL data format.

3255 From the Bus Free Condition, the Master issues a START by driving the SDA line Low while keeping the
 3256 SCL line High. It then issues the Broadcast Address (7'h7E) followed by RnW (0 for Write). Then the Master
 3257 turns on a pull-up resistor and goes to Open Drain. All Slaves ACK by pulling SDA Low (in the Figure, pink
 3258 fill means the Slaves are in control of SDA at this time). The Master then issues the Broadcast Common
 3259 Command Code for ENTHDR2 (0x22) followed by parity bit 'T' (odd parity = 1 for 0x22). At this point, the
 3260 Bus is in HDR-TSL Mode. In this protocol, the SCL line is more than just the clock. Both SDA transitions
 3261 and SCL transitions carry data. The ternary conversion can be understood by considering that each octal digit
 3262 is equivalent to two ternary Symbols. For example, for octal sequence 412255 we have: 4=3+1, 1=0+1,
 3263 2=0+2, 2=0+2, 5=3+2, 5=3+2. The clock pulse occurs when SCL transitions, when SDA transitions, or when
 3264 both lines transition simultaneously.

3265 HDR-TSL Mode is backwards compatible with Legacy I²C Devices, because the High time of an SCL pulse
 3266 is always less than 50ns, and as a result SCL will always appear to be Low due to the I2C 50ns Spike Filter.
 3267 Note that in the Figure, the thick blue High-to-Low transitions of SCL are dummy edges added in order to
 3268 make sure that SCL pulses are always less than 50ns. High-to-Low SCL transitions that occur without a
 3269 simultaneous SDA transition are always dummy transitions.



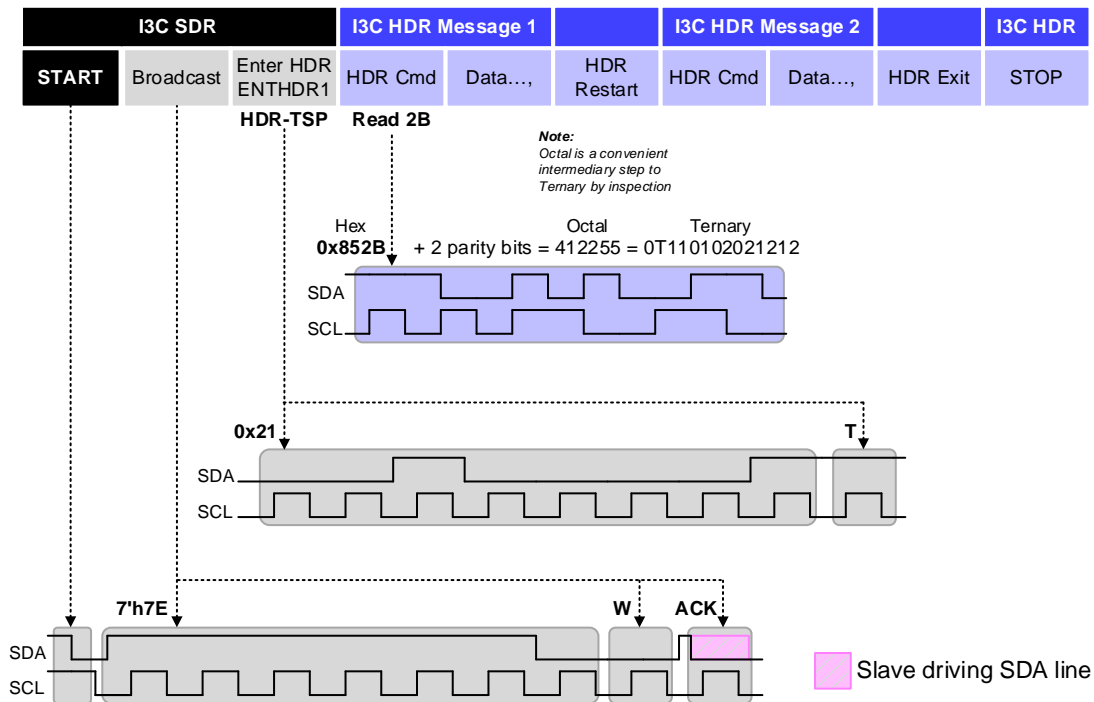
3270
3271

Figure 100 Example Communication Using HDR-TSL Protocol

3272 **Figure 101** illustrates use of HDR-TSP (High Data Rate – Ternary Symbol for Pure Bus) Mode. This protocol
 3273 is only used when there are no I²C Devices on the Bus. The Figure shows how the Master can change the
 3274 Mode from SDR (Single Data Rate) to HDR-TSP Mode, and a sample of the TSL data format.

3275 From the Bus Free Condition, the Master issues a START by driving the SDA line Low while keeping the
 3276 SCL line High. The Master then issues the Broadcast Address (7'h7E) followed by RnW (0 for Write). Then
 3277 the Master turns on a pull-up resistor and goes to Open Drain. All Slaves ACK by pulling SDA Low (in the
 3278 Figure, pink fill means the Slaves are in control of SDA at this time). The Master then issues the Broadcast
 3279 Common Command Code for ENTHDR1 (0x21) followed by parity bit 'T' (odd parity = 1 for 0x21). At this
 3280 point, the Bus is in HDR-TSP Mode. In the HDR-TSP protocol, the SCL is more than just the clock. Both
 3281 SDA transitions and SCL transitions carry data. The ternary conversion can be understood by considering
 3282 that each octal digit is equivalent to two ternary Symbols. For example, for octal sequence 412255 we have:
 3283 4=3+1, 1=0+1, 2=0+2, 2=0+2, 5=3+2, 5=3+2). The clock pulse occurs when SCL transitions, when SDA
 3284 transitions, or when both lines transition simultaneously.

3285 HDR-TSP Mode supports the highest data-rate and uses the least energy per byte, however it is not backwards
 3286 compatible with Legacy I²C Devices since the High duration of an SCL pulse can exceed 50ns.



3287
3288

Figure 101 Example Communication Using HDR-TSP Protocol

Participants

- 1 The list below includes those persons who participated in the Working Group that developed this
- 2 Specification and who consented to appear on this list.

Yossi Amon, Qualcomm Incorporated
Miguel Barasch, Qualcomm Incorporated
Rajesh Bhaskar, Intel Corporation
Geraud Cheenne, STMicroelectronics
Sriraman Dakshinamurthy, InvenSense, Inc.
Kenneth Foust, Intel Corporation
Chris Grigg, MIPI Alliance (staff)
Anubhav Gupta, Intersil Corporation
Hsiehchang Ho, MediaTek Inc.
Doug Hoffman, Qualcomm Incorporated
Toshihisa Hyakuda, Sony Corporation
Paul Kimelman, NXP Semiconductors
Michael Laisne, Dialog Semiconductor
Alessandro Mecchia, STMicroelectronics
Pratap Neelasheety, Synopsys, Inc.
Alex Passi, Cadence Design Systems, Inc.
Venkata Suresh Perumalla, NVIDIA
Radu Pitigoi-Aron, Qualcomm Incorporated

Duane Quiet, Intel Corporation
James Rippie, MIPI Alliance (staff)
Jim Ross, Skyworks Solutions, Inc.
Volker Rzehak, Texas Instruments Incorporated
Brad Sharpe-Geisler, Lattice Semiconductor Corp.
Satwant Singh, Lattice Semiconductor Corp.
Dong Hyun Song, SK Hynix
Amit Srivastava, Intel Corporation
Przemyslaw Sroka, Cadence Design Systems, Inc.
Tatsuya Sugioka, Sony Corporation
Hiroo Takahashi, Sony Corporation
Asmah Truky, Intel Corporation
Suresh Venkatachalam, Synopsys, Inc.
Girish Venkatraman, Intel Corporation
Niel Warren, Knowles Electronics
Rick Wietfeldt, Qualcomm Incorporated
Tim White, IDT