Design Gateway Co.,Ltd
E-mail: ip-sales@design-gateway.com
URL: design-gateway.com

Core Facts

Provided with Core	
Documentation	Reference Design Manual Demo Instruction Manual
Design File Formats	Encrypted File
Instantiation Templates	VHDL
Reference Designs & Application Notes	Vivado Project See Reference Design Manual
Additional Items	Demo on VCK190
Support	
Support Provided by Design Gateway Co., Ltd.	

Features

- NVMe host controller for access one NVMe Gen4 SSD without CPU and external memory
- Support up to four users to access one SSD simultaneously
- Two data buffer modes:
  High speed (1 MB RAM per user) or Small memory (256 KB RAM per user)
- Simple user interfaces by dgIF typeS
- Command support:
  - User#0 (Main user): Identify, Shutdown, Write, Read, SMART, and Flush
  - User#1-#3 (Sub user): Write and Read
- Supported NVMe device
  - Base Class Code:01h (mass storage), Sub Class Code:08h (Non-volatile), and Programming Interface:02h (NVMHCI)
  - MPSMIN (Memory Page Size Minimum): 0 (4 Kbyte)
  - MDTS (Maximum Data Transfer Size): At least 5 (128 Kbyte) or 0 (no limitation)
  - LBA unit: 512 bytes or 4096 bytes
  - Support more than or equal to 2-4 queues
- User clock frequency: More than or equal to PCIe clock (250MHz for Gen4)
- PCIe Hard IP: Integrated Block for PCI Express from Xilinx (256-bit bus interface of 4-lane Gen4)
- Connect to one NVMe SSD directly (without PCIe switch)
- Available reference designs: VCK190 board with AB18-PCIeX16 adapter board
- Customized service for following features
  - Additional NVMe commands such as Format, Write Zeroes, and Sanitize
  - Increase user channel

Table 1: Example Implementation Statistics (Versal)

Family	Example Device	User Count	Buf Mode	Fmax (MHz)	CLB Regs	CLB LUTs	Slice[1]	BRAM Tile	URAM	Design Tools
Versal AI Core	XCVC1902-VSVA2197-2MP-E-S	4	1 MB	375	12387	8138	2505	8	128	Vivado2022.1
		4	256 KB	375	12302	8208	2109	8	32	
		2	1 MB	375	7455	4925	1399	8	64	
		2	256 KB	375	7424	4906	1387	8	16	

Notes: Actual logic resource dependent on percentage of unrelated logic

Applications

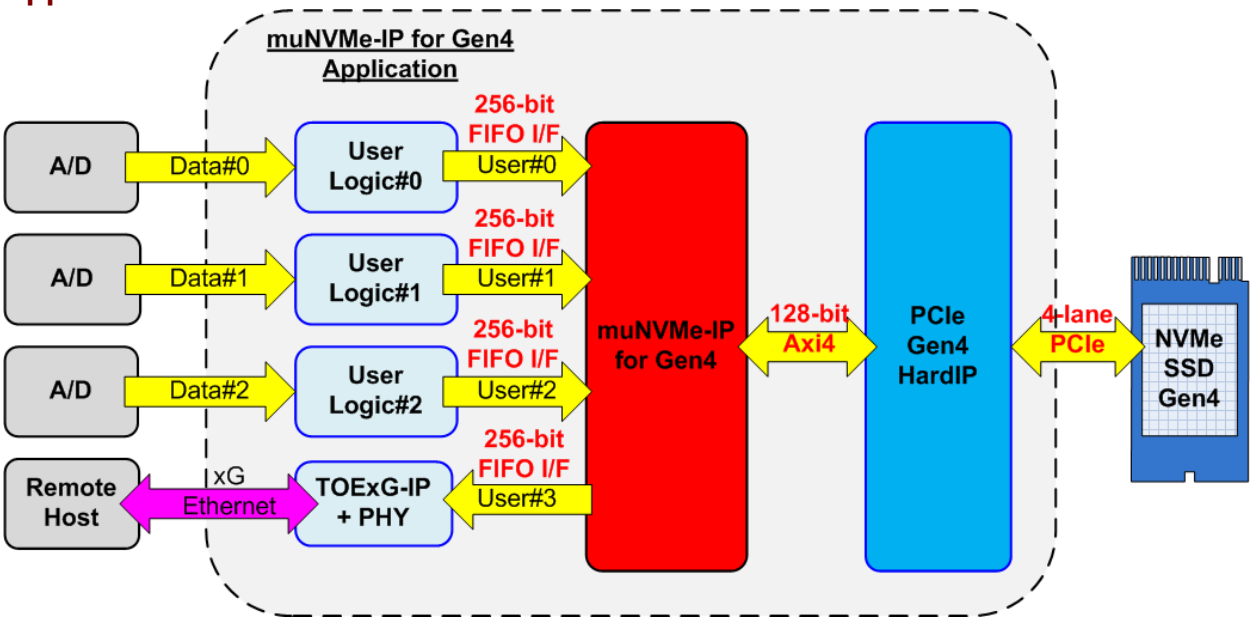

Figure 1: muNVMe IP for Gen4 Application

muNVMe IP Core for Gen4 integrated with PCIe Gen4 hard IP (Integrated Block for PCI Express) inside Xilinx FPGA is ideal to access NVMe Gen4 SSD without CPU and external memory such as DDR. In accordance with four-user interface, all users are allowed to access the same NVMe SSD parallelly. Thus, this solution fits the application which needs to record many data types from several sources such as high-speed A/D converter to the same NVMe SSD. Meanwhile, there is a user reading the data from the SSD for monitoring the operation. When an Ethernet IP Core is connected to the read logic, the remote host can be adapted for reading the data and monitoring data progress across Ethernet network, as shown in Figure 1.

General Description

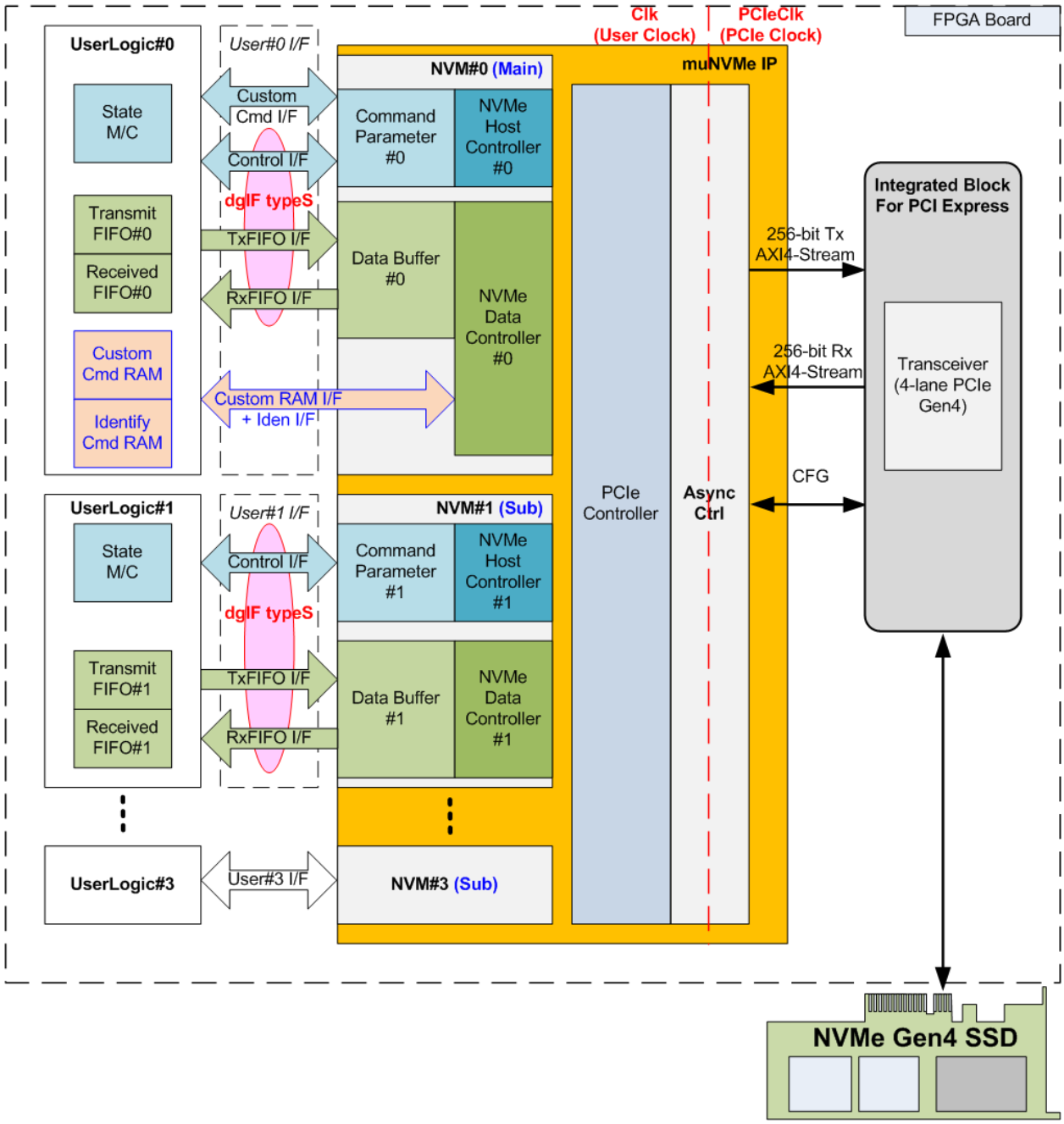

Figure 2: muNVMe IP for Gen4 block diagram

muNVMe IP for Gen4 implements the host controller to access NVMe Gen4 SSD following NVM express standard. Physical interface of NVMe SSD is PCIe which is handled by Integrated Block for PCI Express (PCIe hard IP) in Xilinx FPGA.

muNVMe IP supports six NVMe commands, i.e., Identify, Shutdown, Write, Read, SMART, and Flush command with four user interfaces - User#0-#3 I/F. As shown in Figure 2, User#0-#3 I/F are the interface of NVM#0-#3 module inside muNVMeIP, respectively. All six commands are supported by NVM#0 (Main) while NVM#1-#3 (Sub) support only Write and Read command.

For operating each command, all user I/F consists of two interface groups. First is Control interface for transferring command and the parameters. Another is Data interface for transferring data when the command must have the data transferring. The Control interface and Data interface for Write/Read command use dgIF typeS format. Control interface of dgIF typeS consists of start address and transfer length with asserting the request signal while Data interface of dgIF typeS is the FIFO interface.

SMART and Flush command require the specific interface, named Custom interface, which consists of Ctm I/F for control path and Ctm RAM I/F for data path. Furthermore, Identify command has its own data interface, named Iden RAM I/F, as shown in Figure 2. All of these specific interfaces are integrated only in User#0 I/F.

Due to four user interfaces, while some NVMs (NVM#0-#3) operating the command, the remaining NVMs can also operate other commands parallelly. However, there is one limitation about the parallel operation. Shutdown command must be executed on NVMe#0 after all NVMs return to be idle (no Write/Read command operating on NVM#1-#3).

While initializing or running some commands, error signal may be asserted by muNVMe IP if some abnormal conditions are found. The IP includes the error status to check more details. The error recovery is done by asserting reset to muNVMe IP and SSD.

In order to transmit packets continuously to the PCIe hard IP until the end of the packet, the User Clock frequency must meet a certain requirement. The transmitted data must remain valid from the start to the end of the packet. To fulfill this requirement, the User Clock frequency must be equal to or greater than the PCIe Clock frequency, which is set at 250 MHz for 4-lane PCIe Gen4. This will ensure that the bandwidth of the transmit logic is sufficient to meet the bandwidth of the PCIe hard IP.

The reference designs on FPGA evaluation boards are available for evaluation before purchasing.

Functional Description

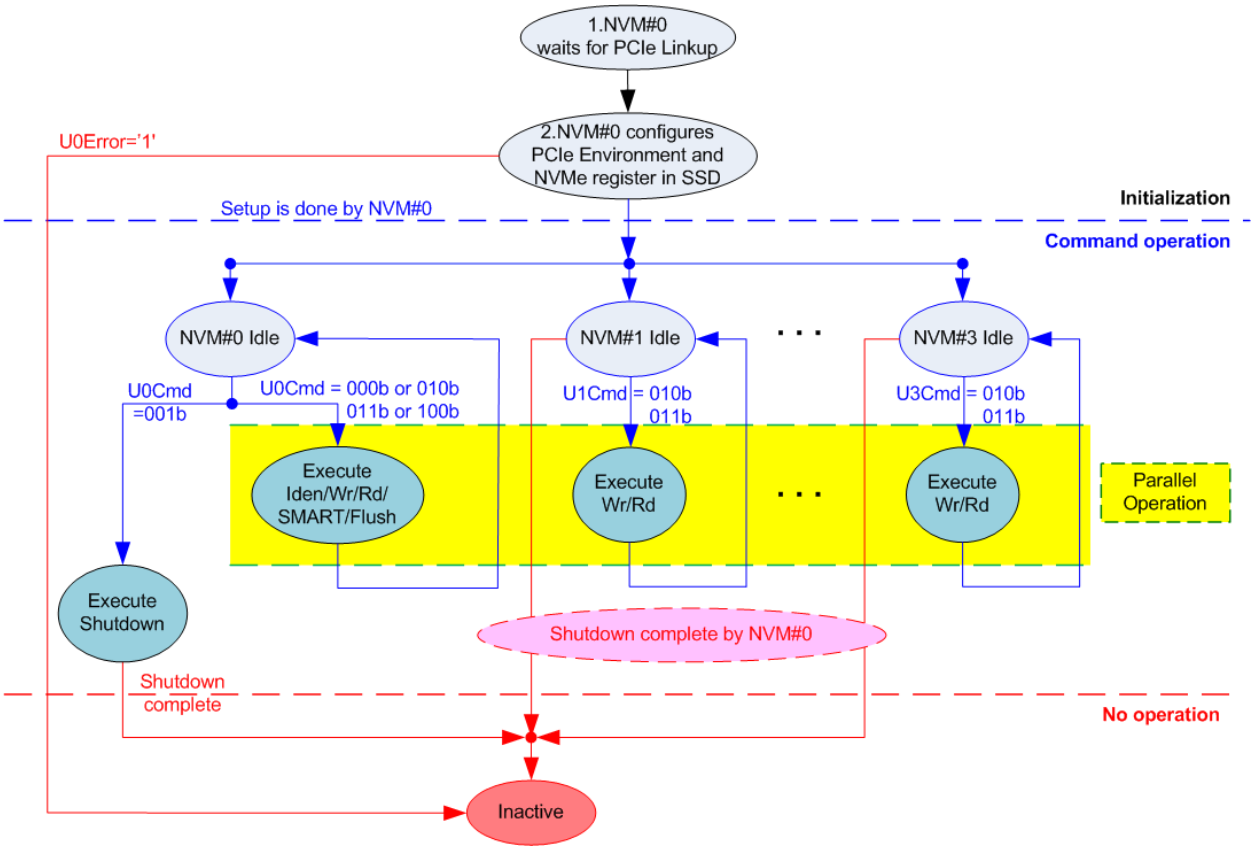

Figure 3: Operation Flow of muNVMe IP

The operation flow of muNVMe IP is divided into three phases, i.e., Initialization, Command operation, and No operation as shown in Figure 3. Firstly, the Initialization phase is handled by User#0 I/F. In this phase, User#1-3 I/F merely waits until the end of Initialization. The Initialization step includes

1) NVM#0 waits until PCIe is ready by monitoring Linkup status from PCIe IP core.
2) NVM#0 begins the initialization process by setup PCIe registers and NVMe registers. If some errors are detected during setup process, NVM#0 changes the IP to the Inactive state.

After the initialization process is done, all user interfaces are idle and ready to receive command from user. The command of each User I/F can be requested parallelly except Shutdown command. All users must return to Idle before Shutdown command is requested through User#0 I/F. After the Shutdown is done, all NVMs change to the Inactive state.

.

More details of each command operation flow in Command operation phase of NVM#0 are shown in Figure 4.

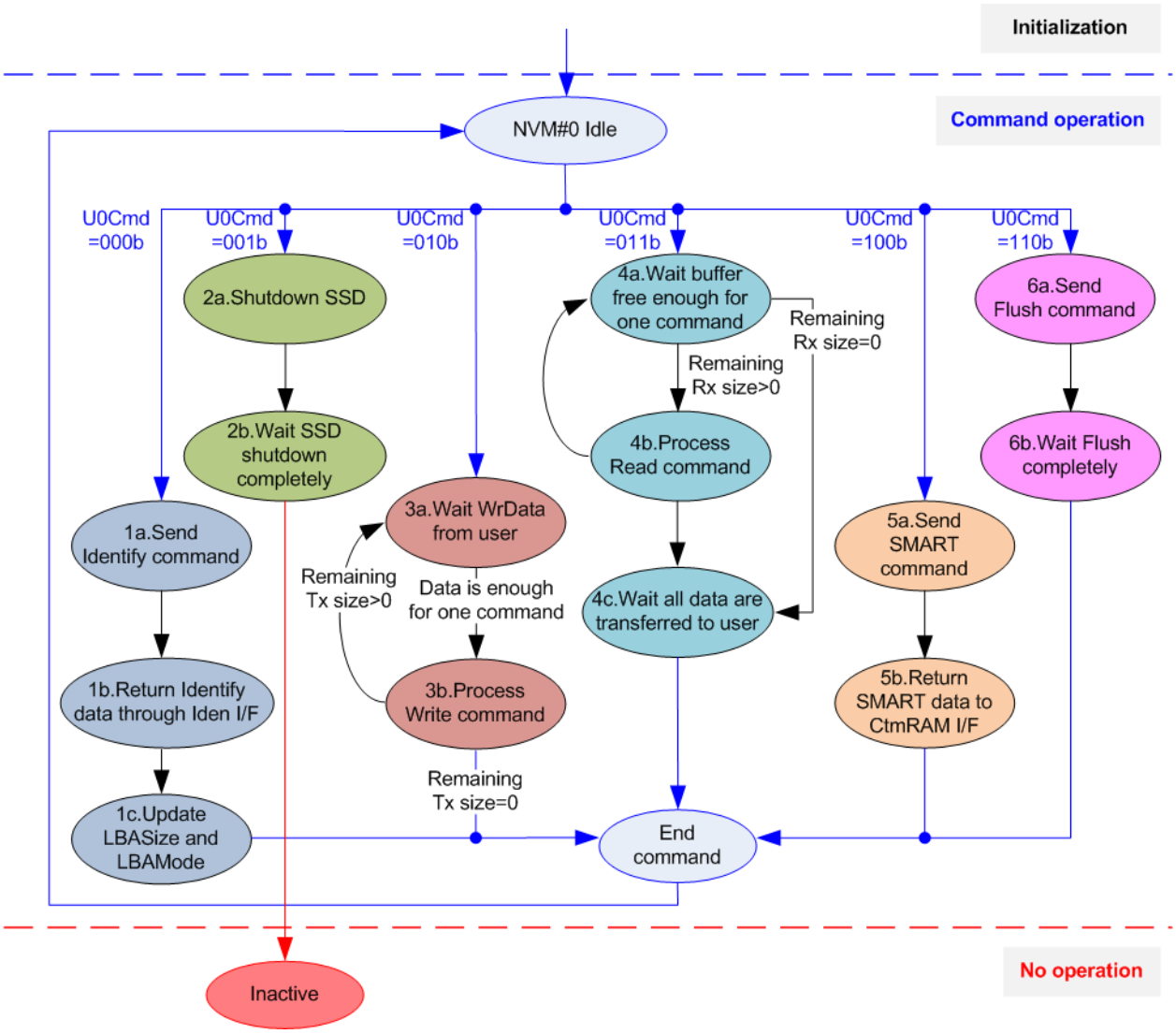

Figure 4: Command Operation Flow of NVM#0, controlled by User#0 I/F

1) The 1st command from user must be Identify command (U0Cmd=000b) to update LBASize (disk capacity) and LBAMode (LBA unit=512 bytes or 4 Kbytes).
2) The last command before power down the system must be Shutdown command (U0Cmd=001b). This command is recommended to guarantee SSD powered down in a good sequence. Without Shutdown command, Write data in SSD cannot be guaranteed. After Shutdown command is done, muNVMe IP and SSD enter to the Inactive state and do not receive any commands until the IP is reset.

3) When Write command is requested to NVM#0 (U0Cmd=010b), the maximum data size for one command is 128 Kbytes. If total length from user is more than 128 Kbytes, repeat step 3a) – 3b) for transferring more data.
   a) The IP waits until Write data, sent by user, is enough for one command (transfer size of one command in muNVMe IP is 128 Kbytes, except the last loop which could be less than 128 Kbytes).
   b) The IP sends Write command to SSD and then waits until the status is returned from SSD. NVM#0 status returns to the Idle state when total data is completely transferred. Otherwise, NVM#0 returns to step 3a) to send the next Write command.
4) Similar to Write command, Read command (U0Cmd=011b) must be sent by NVM#0 to the SSD many times when total length from user is more than 128 Kbytes. As the result, step 4a) – 4b) are repeated until all data are requested.
   a) The IP waits until free space of Data Buffer#0 in muNVMe IP is enough for one command. If remaining transfer size is equal to zero, the IP skips to step 4c).
   b) The IP sends Read command to SSD. After that, returns to step 4a) to check the remaining transfer size and the free space of Data Buffer#0.
   c) IP waits until all data are completely transferred from Data Buffer#0 to UserLogic and then changes NVM#0 status back to the Idle state. Therefore, Data Buffer of NVM#0 is empty after Read command is done.
5) For SMART command (U0Cmd=100b), 512-byte data is returned after the operation is done.
   a) IP sends Get Log Page command to read SMART/Health information from SSD.
   b) 512-byte data is returned from the SSD. The IP forwards the data through Custom command RAM interface (CtmRamAddr=0x000 – 0x01F).
6) For Flush command (U0Cmd=110b), there is no data transferring during the operation.
   a) IP sends Flush command to SSD.
   b) IP waits until SSD returns status to complete the operation.

For Write command (U1Cmd=010b) and Read command (U1Cmd=011b) of NVM#1-#3, the operation flow is completely the same as Write and Read command of NVM#0 by using separated operation core and Data Buffer of NVM#1-#3.

To design NVMe host controller, muNVMe IP implements two protocols, i.e., NVMe protocol for interface with user and PCIe protocol for interface with PCIe hard IP. Therefore, the hardware inside muNVMe-IP is divided to NVMe block and PCIe block. The details of the hardware inside muNVMe IP are described as follows.

NVMe

muNVMe IP supports six commands, i.e., Identify, Write, Read, SMART, Flush, and Shutdown command which can split in two command types, i.e., Admin command and NVM command. muNVMe IP supports three Admin commands, i.e., Identify, Shutdown, and SMART command and supports three NVM commands, i.e., Write, Read, and Flush command. After the command operation is done, the status returned from the SSD is latched to U0AdmCompStatus signal when running Admin command or U0-U3IOCompStatus signal when running NVM command.

The parameters of Write or Read command are set by Control interface of dgIF typeS while the parameters of SMART or Flush command are set by CtmSubm of Ctm interface. Data interface for Write or Read command is transferred by FIFO interface, a part of dgIF typeS, which is finally connected with the Data Buffer inside the each NVM. The data interface of other commands has its own interface, i.e., Identify RAM for Identify command and Custom RAM for SMART command.

To implement NVMe protocol, all NVMs are designed to have four NVMe submodules, i.e., NVMe Host Controller, Command Parameter, Data Buffer, and NVMe Data Controller. However, submodule features and complexity between NVM#0 and others NVM are not exactly identical because of different user command support. The details of each submodule are described as follows.

- **NVMe Host Controller**

  NVMe Host Controller is the core controller of each NVM. On the other word, it is the main controller of muNVMe IP. The module operation is split into two phases. First is the Initialization phase which is once run after the system is boot up for setting NVMe register inside the SSD. The initialization is operated only in NVMe Host Controller#0 as described early in Figure 4. After finishing the Initialization phase, the next phase is the Command operation phase. The order of transmitted packet and received packet in each user interface is controlled by its NVMe Host Controller.

  To operate each command, the parameters of the command are latched to Command Parameter for creating the packet. After that, the packet is forwarded to AsyncCtrl to convert NVMe packet to PCIe packet. After each command operation is done, the status packet is returned from SSD. The Host Controller decodes the status value and check whether the operation is complete or error. If the command needs to transfer the data such as Write command and Read command, the Host Controller must handle the order of data packet that is created and decoded by the Data controller.

- **Command Parameter**

  This module prepares the Command packet sent to the SSD that uses the parameter set by the user. Also, the status value is extracted from the Status packet which is returned from the SSD. Typically, the command consists of 16 Dwords (1 Dword = 32-bit). When running Identify, Shutdown, Write, or Read command, all 16 Dwords are prepared by using the user inputs on dgIF typeS. When running SMART and Flush command, all 16 Dwords are directly loaded via CtmSubmDW0-CtmSubmDW15 of Ctm interface.

- **Data Buffer**

  Two data buffer modes are supported, i.e., High speed mode which uses 1 Mbyte RAM and Small memory mode which uses 256 Kbyte RAM. The RAM is implemented by using BlockRAM. The buffer stores data for transferring with the SSD while operating Write and Read command.

- **NVMe Data Controller**

  This module is operated when the command must transfer the data, i.e., Identify, SMART, Write, and Read command. There are three data interfaces for transferring with the SSD, i.e., FIFO interface with the Data Buffer when running Write or Read command, Custom command RAM interface when running SMART command, and Identify interface when running Identify command. The address of the Data Buffer is controlled by this module when running Write or Read command.

PCIe

The PCIe standard is the outstanding low-layer protocol for very high-speed application. The NVMe standard is the protocol which is run over PCIe protocol. In the initialization process, NVMe layer is setup after finishing PCIe layer setup. Two modules are designed to support PCIe protocol - PCIe controller and AsyncCtrl. More details of each module are described as follows.

- **PCIe Controller**

  In initialization process, PCIe Controller sets up PCIe environment of SSD via CFG interface. After that, PCIe packet is created or decoded via 256-bit Tx/Rx AXI4-Stream. The command packet and the data packet from NVMe module are converted to be PCIe packet by PCIe Controller and vice versa.

- **AsyncCtrl**

  AsyncCtrl includes asynchronous registers and asynchronous buffers with asymmetric data width to support clock domain crossing. Most logics in muNVMe IP run on user clock domain while PCIe hard IP runs on PCIe clock domain. The AXI4-Stream interface of the PCIe hard IP must consistently transmit each packet's data. Consequently, the frequency of the User Clock must be set at a level that is equal to or higher than that of the PCIe Clock to ensure that the user interface bandwidth is equal to or greater than the PCIe interface bandwidth.

User Logic

This module could be designed by using small state machine to send the commands and the parameters for each command. For example, the address and transfer size which are the parameters for Write or Read command are designed by using simple registers. While FIFO is connected for transferring data in Write and Read command. The data output of SMART command and Identify command interface connects to simple dual port RAM with byte enable. The data width of FIFO and RAM are 256-bit while the memory depth can be set by different value. Data size of Identify command is 8 Kbytes while data size of SMART command is 512 bytes.

Integrated Block for PCI Express

Some UltraScale+ devices and Versal devices have Integrated Block for PCI Express (PCIe hard IP) to operate PCIe Gen4 protocol, called PCIe4C and PL PCIE4. To connect with muNVMe IP, PCIe4C or PL PCIE4 is configured to be 4-lane PCIe Gen4 by using 256-bit data interface. Each muNVMe IP uses one PCIe hard IP for controlling one NVMe Gen4 SSD. Therefore, the maximum number of SSDs connecting to one FPGA device is limited by the number of PCIe hard IPs in FPGA device.

The IP wizard on Xilinx tool to generate PCIe hard IP when selecting UltraScale+ and Versal device are different. On UltraScale+ device, the user uses one IP wizard to generate the PCIe hard IP integrating with the transceiver. On Versal device, the user needs to call two IP wizards to generate PCIe hard IP and the Transceiver individually.

More details of PCIe hard IP are described in following document

PG213: UltraScale+ Devices Integrated Block for PCI Express
https://www.xilinx.com/products/intellectual-property/pcie4-ultrascale-plus.html#documentation
PG343: Versal ACAP Integrated Block for PCI Express
https://www.xilinx.com/products/intellectual-property/pcie-versal.html#documentation

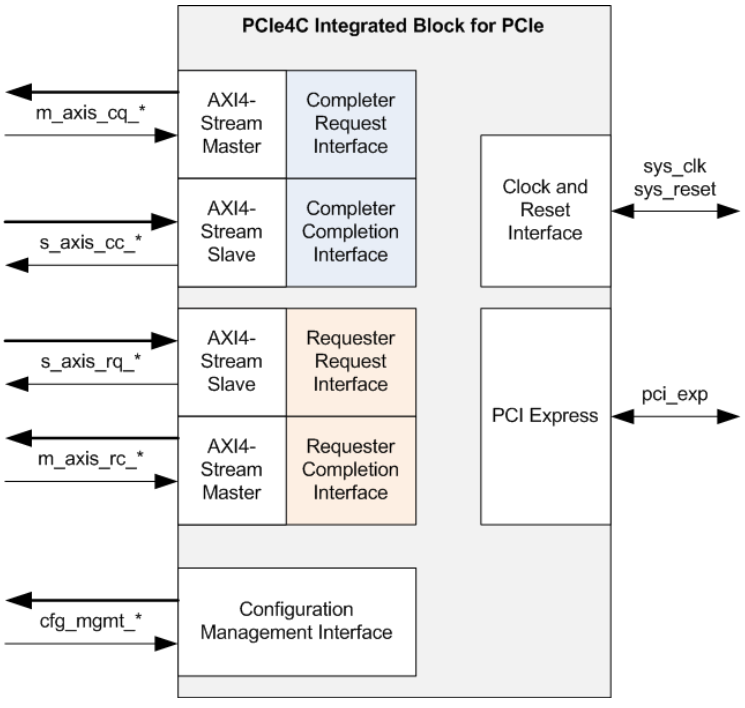

Figure 5: PCI4C Integrated Block for PCI Express

Core I/O Signals

Descriptions of all signal I/O are provided in Table 2 **–** Table 4.

Table 2: Core Parameters

Name	Value	Description
UserCount	2 - 4	The number of User I/F. U0 I/F and U1 I/F are always available. U2 I/F is available when setting UserCount=3 or 4. U3 I/F is available when setting UserCount=4. When there are some User I/Fs unavailable, the I/O signals of the unavailable ports are shown but there is no operation on them.
BufMode	0 or 1	Data buffer mode. 1-High speed mode by using 1 MB buffer, 0-Small memory mode by using 256 KB buffer

Table 3: User logic I/O Signals (Synchronous to Clk signal)

Signal	Dir	Description
Control I/F of dgIF typeS		
RstB	In	Synchronous reset signal. Active low. De-assert to '1' when Clk signal is stable.
Clk	In	User clock for running muNVMe IP. The frequency must be more than or equal to PCIeClk frequency, clock output from PCIe Hard IP. (250 MHz for PCIe Gen4).
U0Cmd[2:0]	In	User Command of User#0 I/F. Valid when U0Req='1'. 000b: Identify, 001b: Shutdown, 010b: Write SSD, 011b: Read SSD, 100b: SMART, 110b: Flush, Others: Reserved
U1-U3Cmd[2:0]	In	User Command of User#1-#3 I/F. Valid when U1-U3Req='1', respectively. 010b: Write SSD, 011b: Read SSD, Others: Reserved
U0-U3Addr[47:0]	In	Start address of User#0-#3 I/F to write/read SSD in 512-byte unit. Valid when U0-U3Req='1', respectively. In case LBA unit = 4 Kbyte, U0-U3Addr[2:0] must be always set to 000b to align 4-Kbyte unit. In case LBA unit = 512 byte, it is recommended to set U0-U3Addr[2:0]=000b to align 4-Kbyte size (SSD page size). Write/Read performance of most SSDs is reduced when start address is not aligned to page size.
U0-U3Len[47:0]	In	Total transfer size to write/read SSD in 512-byte unit. Valid from 1 to (LBASize - U0-U3Addr). In case LBA unit = 4 Kbyte, U0-U3Len[2:0] must be always set to 000b to align 4-Kbyte unit. Valid when U0-U3Req='1', respectively.
U0Req	In	Assert to '1' to send the new command request and de-assert to '0' after IP starts the operation by asserting U0Busy to '1'. This signal can be asserted when NVM#0 is Idle (U0Busy='0'). Command parameters (U0Cmd, U0Addr, U0Len, and CtmSubmDW0-DW15) must be valid and stable when U0Req='1'. U0Addr and U0Len are inputs for Write/Read command while CtmSubmDW0-DW15 are inputs for SMART/Flush command.
U1-U3Req	In	Assert to '1' to send the new command request and de-assert to '0' after IP starts the operation by asserting U1-U3Busy to '1'. This signal can be asserted when NVM#1-#3 is Idle (U1-U3Busy='0'). Command parameters (U1-U3Cmd, U1-U3Addr, and U1-U3Len) must be valid and stable when U1-U3Req='1', respectively.
U0-U3Busy	Out	Asserted to '1' when NVM#0-#3 is busy (operate command of User#0-#3 I/F), respectively. New request must not be sent (U0-U3Req to '1') when User#0-#3 is busy, respectively.

Signal	Dir	Description
Control I/F of dgIF typeS		
LBASize[47:0]	Out	Total capacity of SSD in 512-byte unit. Default value is 0. This value is valid after finishing Identify command.
LBAMode	Out	LBA unit size of SSD ('0': 512 bytes, '1': 4 Kbytes). Default value is 0. This value is valid after finishing Identify command.
U0-U3Error	Out	Error flag. Asserted to '1' when U0-U3ErrorType is not equal to 0. The flag can be cleared by asserting RstB to '0'.
U0ErrorType[31:0]	Out	[0] – Error when PCIe class code is not correct. [1] – Error from CAP (Controller capabilities) register which may be caused from - MPSMIN (Memory Page Size Minimum) is not equal to 0. - NVM command set flag (bit 37 of CAP register) is not set to 1. - DSTRD (Doorbell Stride) is not 0. - MQES (Maximum Queue Entries Supported) is less than 16. More details of each register can be checked from NVMeCAPReg signal. [2] – Error when Admin completion entry is not received until timeout. [3] – Error when status register in Admin completion entry is not 0 or phase tag/command ID is invalid. Please see more details from U0AdmCompStatus signal. [4] – Error when IO completion entry of NVM#0 is not received until timeout. [5] – Error when status register in IO completion entry of NVM#0 is not 0 or phase tag is invalid. Please see more details from U0IOCompStatus signal. [6] – Error from unsupported LBA unit (LBA unit is not equal to 512 bytes or 4 Kbytes) [7] – Error when SSD does not support multiple queues. [8] – Error when receiving TLP packet with incorrect size [9] – Error when PCIe hard IP detects Error correction code (ECC) error from the internal buffer Bit[15:10] are mapped to Uncorrectable Error Status Register [10] – Mapped to Unsupported Request Error Status (bit[20]) [11] – Mapped to Completer Abort Status (bit[15]) [12] – Mapped to Unexpected Completion Status (bit[16]) [13] – Mapped to Completion Timeout Status (bit[14]) [14] – Mapped to Poisoned TLP Received Status (bit[12]) [15] – Mapped to ECRC Error Status (bit[19]) [23:16] - Reserved Bit[30:24] are also mapped to Uncorrectable Error Status Register [24] – Mapped to Data Link Protocol Error Status (bit[4]) [25] – Mapped to Surprise Down Error Status (bit[5]) [26] – Mapped to Receiver Overflow Status (bit[17]) [27] – Mapped to Flow Control Protocol Error Status (bit[13]) [28] – Mapped to Uncorrectable Internal Error Status (bit[22]) [29] – Mapped to Malformed TLP Status (bit[18]) [30] – Mapped to ACS Violation (bit[21]) *Note: Timeout period of bit[2]/[4] is set from TimeOutSet input*
U1-U3ErrorType[31:0]	Out	[3:0] – Reserved [4] – Error when IO completion entry of NVM#1-#3 respectively is not received until timeout. [5] – Error when status register in IO completion entry of NVM#1-#3 respectively is not 0 or phase tag is invalid. Please see more details from U1-U3IOCompStatus signal. [31:6] – Reserved *Note: Timeout period of bit[4] is set from TimeOutSet input.*

Signal	Dir	Description
Data I/F of dgIF typeS		
U0FifoWrCnt[15:0]	In	Write data counter of Receive FIFO#0. Used to check full status. If FIFO data counter signal is less than 16 bits, please fill '1' to upper bit.
U0FifoWrEn	Out	Asserted to '1' to write data to Receive FIFO#0 when running Read command with NVM#0.
U0FifoWrData[255:0]	Out	Write data bus of Receive FIFO#0. Valid when U0FifoWrEn='1'.
U0FifoRdCnt[15:0]	In	Read data counter of Transmit FIFO#0. Used to check data size stored in FIFO. If FIFO data counter signal is less than 16 bits, please fill '0' to upper bit.
U0FifoEmpty	In	The signal is unused for this IP.
U0FifoRdEn	Out	Asserted to '1' to read data from Transmit FIFO#0 when running Write command with NVM#0.
U0FifoRdData[255:0]	In	Read data returned from Transmit FIFO#0. Valid in the next clock after U0FifoRdEn='1'.
Data I/F for User#1-#3 use the same description as User#0 I/F by referring to signal and module of NVM#1-#3. User#1-#3 Data I/F signals consist of U1-U3FifoWrCnt[15:0], U1-U3FifoWrEn, U1-U3FifoWrData[255:0], U1-U3FifoRdCnt[15:0], U1-U3FifoEmpty, U1-U3FifoRdEn, and U1-U3FifoRdData[255:0], respectively.		
muNVMe IP for Gen4 Interface		
IPVesion[31:0]	Out	IP version number
U0TestPin[47:0], U1-U3TestPin[15:0]	Out	Reserved to be IP Test point.
TimeOutSet[31:0]	In	Timeout value to wait completion from SSD. Time unit is equal to 1/(Clk frequency). When TimeOutSet is set to 0, Timeout function is disabled.
U0AdmCompStatus[15:0]	Out	Status output from Admin Completion Entry [0] – Set to '1' when Phase tag or Command ID in Admin Completion Entry is invalid. [15:1] – Status field value of Admin Completion Entry
U0-U3IOCompStatus[15:0]	Out	Status output from IO Completion Entry [0] – Set to '1' when Phase tag in IO Completion Entry of NVM#0-#3 is invalid, respectively. [15:1] – Status field value of IO Completion Entry in NVM#0-#3, respectively.
NVMeCAPReg[31:0]	Out	The parameter value of the NVMe capability register when U0ErrorType[1] is asserted to '1'. [15:0] – MQES (Maximum Queue Entries Supported) [19:16] – DSTRD (Doorbell Stride) [20] – NVM command set flag [24:21] – MPSMIN (Memory Page Size Minimum) [31:25] – Undefined
Identify Interface (for User#0)		
IdenWrEn	Out	Asserted to '1' for sending data output from Identify command.
IdenWrDWEn[7:0]	Out	Dword (32-bit) enable of IdenWrData. Valid when IdenWrEn='1'. '1': This Dword data is valid, '0': This Dword data is not available. Bit[0], [1], [2], …, [7] corresponds to IdenWrData[31:0], [63:32], … , [255:224] respectively.
IdenWrAddr[7:0]	Out	Index of IdenWrData in 256-bit unit. Valid when IdenWrEn='1'. 0x00-0x7F is 4Kbyte Identify controller data, 0x80-0xFF is 4Kbyte Identify namespace data.
IdenWrData[255:0]	Out	4Kbyte Identify controller data or Identify namespace data. Valid when IdenWrEn='1'.

Signal	Dir	Description
Custom Interface (for User#0)		
CtmSubmDW0[31:0] – CtmSubmDW15[31:0]	In	16 Dwords of Submission queue entry for SMART/Flush command. DW0: Command Dword0, DW1: Command Dword1, …, and DW15: Command Dword15. These inputs must be valid when U0Req='1' and U0Cmd=100b (SMART) or 110b (Flush).
CtmCompDW0[31:0] – CtmCompDW3[31:0]	Out	4 Dwords of Completion queue entry, output from SMART/Flush command. DW0: Completion Dword0, DW1: Completion Dword1, …, and DW3: Completion Dword3
CtmRamWrEn	Out	Asserted to '1' for sending data output from Custom command such as SMART command.
CtmRamWrDWEn[7:0]	Out	Dword (32-bit) enable of CtmRamWrData. Valid when CtmRamWrEn='1'. '1': This Dword data is valid, '0': This Dword data is not available. Bit[0], [1], …, [7] corresponds to CtmRamWrData[31:0], [63:32], …, [255:224], respectively.
CtmRamAddr[7:0]	Out	Index of CtmRamWrData when SMART data is received. Valid when CtmRamWrEn='1'. (Optional) Index to request data input through CtmRamRdData for customized Custom commands.
CtmRamWrData[255:0]	Out	512-byte data output from SMART command. Valid when CtmRamWrEn='1'.
CtmRamRdData[255:0]	In	(Optional) Data input for customized Custom commands.

Table 4: Physical I/O Signals for PCIe Gen4 Hard IP (Synchronous to PCIeClk)

Signal	Dir	Description
PCIe Gen4 hard IP		
PCIeRstB	In	Synchronous reset signal. Active low. De-asserted to '1' when PCIe hard IP is not in reset state.
PCIeClk	In	Clock output from PCIe hard IP (250 MHz for PCIe Gen4).
PCIeLinkup	In	Asserted to '1' when PCIe hard IP is linked up.
Configuration Management Interface		
PCIeCfgDone	In	Read/Write operation complete. Assert for 1 cycle when the operation completes.
PCIeCfgRdEn	Out	Read enable. Asserted to '1' for a read operation.
PCIeCfgRdData[31:0]	In	Read data. Valid when PCIeCfgDone is asserted to '1'.
PCIeCfgWrEn	Out	Write enable. Asserted to '1' for a write operation.
PCIeCfgWrData[31:0]	Out	Write data which is used to configure the Configuration and Management registers.
PCIeCfgByteEn[3:0]	Out	Byte enable for write data, where bit[0],[1],[2], and [3] corresponds to PCIeCfgWrData[7:0], [15:8], [23:16], and [31:24], respectively
PCIeCfgAddr[9:0]	Out	Read/Write Address.
Requester Request Interface		
PCIeMtTxData[255:0]	Out	Requester request data bus.
PCIeMtTxKeep[7:0]	Out	Bit [i] indicates that Dword [i] of PCIeMtTxData contains valid data; i=0-7.
PCIeMtTxLast	Out	Asserted this signal in the last cycle of a TLP to indicate the end of the packet.
PCIeMtTxReady[3:0]	In	Assert to accept data. Data is transferred when both PCIeMtTxValid and PCIeMtTxReady are asserted in the same cycle.
PCIeMtTxUser[61:0]	Out	Requester request user data. Valid when PCIeMtTxValid is high.
PCIeMtTxValid	Out	Asserted to drive valid data on PCIeMtTxData bus. muNVMe IP keeps the valid signal asserted during the transfer of packet.

Signal	Dir	Description
Completer Request Interface		
PCIeMtRxData[255:0]	In	Received data from PCIe hard IP.
PCIeMtRxKeep[7:0]	In	Bit [i] indicates that Dword [i] of PCIeMtRxData contains valid data; i=0-7.
PCIeMtRxLast	In	Assert this signal in the last beat of a packet to indicate the end of the packet.
PCIeMtRxReady	Out	Indicates that muNVMe IP is ready to accept data.
PCIeMtRxUser[74:0]	In	Sideband information for the TLP being transferred. Valid when PCIeMtRxValid is high.
PCIeMtRxValid	In	Assert when PCIe hard IP drives valid data on PCIeMtRxData bus. PCIe hard IP keeps the valid signal asserted during the transfer of packet.
Completer Completion Interface		
PCIeSlTxData[255:0]	Out	Completion data from muNVMe IP.
PCIeSlTxKeep[7:0]	Out	Bit [i] indicates that Dword [i] of PCIeSlTxData contains valid data; i=0-7.
PCIeSlTxLast	Out	Asserted this signal in the last cycle of a packet to indicate the end of the packet.
PCIeSlTxReady[3:0]	In	Indicates that PCIe hard IP is ready to accept data.
PCIeSlTxUser[32:0]	Out	Sideband information for the TLP being transferred. Valid when PCIeSlTxValid is high.
PCIeSlTxValid	Out	Asserted to drive valid data on PCIeSlTxData bus. muNVMe IP keeps the valid signal asserted during the transfer of a packet.
Requester Completion Interface		
PCIeSlRxData[255:0]	In	Received data from PCIe hard IP.
PCIeSlRxKeep[7:0]	In	Bit [i] indicates that Dword [i] of PCIeSlRxData contains valid data; i=0-7.
PCIeSlRxLast	In	Assert this signal in the last beat of a packet to indicate the end of the packet.
PCIeSlRxReady	Out	Indicates that muNVMe IP is ready to accept data.
PCIeSlRxUser[87:0]	In	Sideband information for the TLP being transferred. Valid when PCIeSlRxValid is high.
PCIeSlRxValid	In	Assert when PCIe hard IP drives valid data on PCIeSlRxData bus. PCIe hard IP keeps the valid signal asserted during the transfer of packet.

Timing Diagram

Initialization

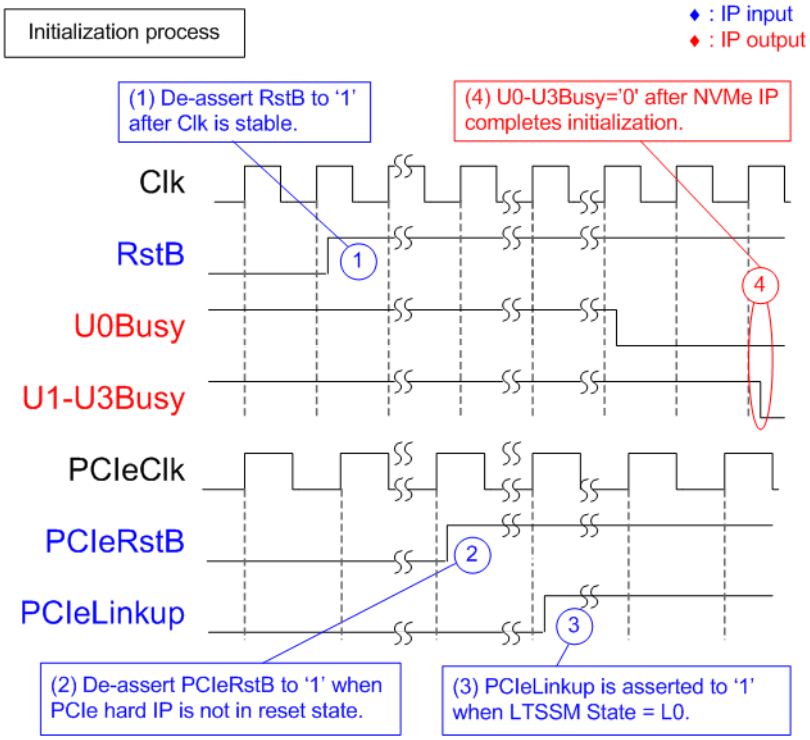

Figure 6: Timing diagram during initialization process

The step of the initialization process is as follows.

1) Wait until Clk is stable and then de-asserts RstB to '1' to start IP initialization.
2) PCIe hard IP de-asserts PCIeRstB to '1' after finishing PCIe reset sequence. PCIe hard IP is ready to transfer data with the application layer.
3) PCIe hard IP asserts PCIeLinkup to '1' after LTSSM state of PCIe hard IP is L0 state. After that, muNVMe IP starts initialization process.
4) U0-U3Busy are de-asserted to '0' after muNVMe IP completes initialization process.
   Note: U1-U3Busy are de-asserted after U0Busy is de-asserted for 2 clock cycles.

After finishing above sequences, muNVMe IP is ready to receive the command from user.

Control interface of dgIF typeS

dgIF typeS signals are split into two groups. First group is Control interface for sending command with the parameters and monitoring the status. Second group is Data interface for transferring data stream in both directions. Figure 7 shows dgIF typeS Control interface of NVM#0.

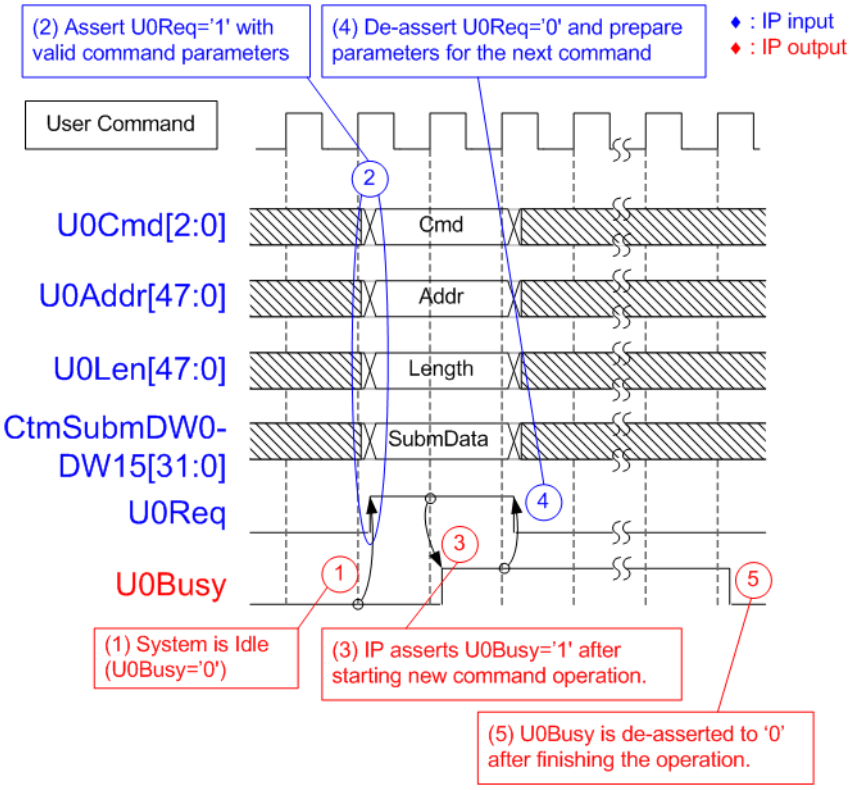

Figure 7: Control interface of dgIF typeS timing diagram

1) Before sending new command request to the IP through User#0 I/F, U0Busy must be equal to '0' to confirm that the desired NVM is Idle.
2) Command and the parameters such as U0Cmd, U0Addr, and U0Len must be valid when asserting U0Req to '1' for sending the new command request.
3) IP asserts U0Busy to '1' after starting the new command operation.
4) After U0Busy is asserted to '1', U0Req is de-asserted to '0' to finish the current request. New parameters for the next command could be prepared on the bus. U0Req for the new command must not be asserted to '1' until the current command operation is done.
5) U0Busy is de-asserted to '0' after the command operation is completed. New command request could be sent by asserting U0Req to '1'.

Note: The number of parameters using in each command is different, described as follows.

- *Write and Read command: Use U0Cmd, U0Addr, and U0Len*
- *SMART and Flush command: Use U0Cmd and CtmSubmDW0-DW15*
- *Identify and Shutdown command: Use only U0Cmd*

Similarly, dgIF typeS Control interface of NVM#1-#3 has the same concept as NVM#0 by using U1-U3Busy, U1-U3Cmd, U1-U3Addr, and U1-U3Len for Write and Read command.

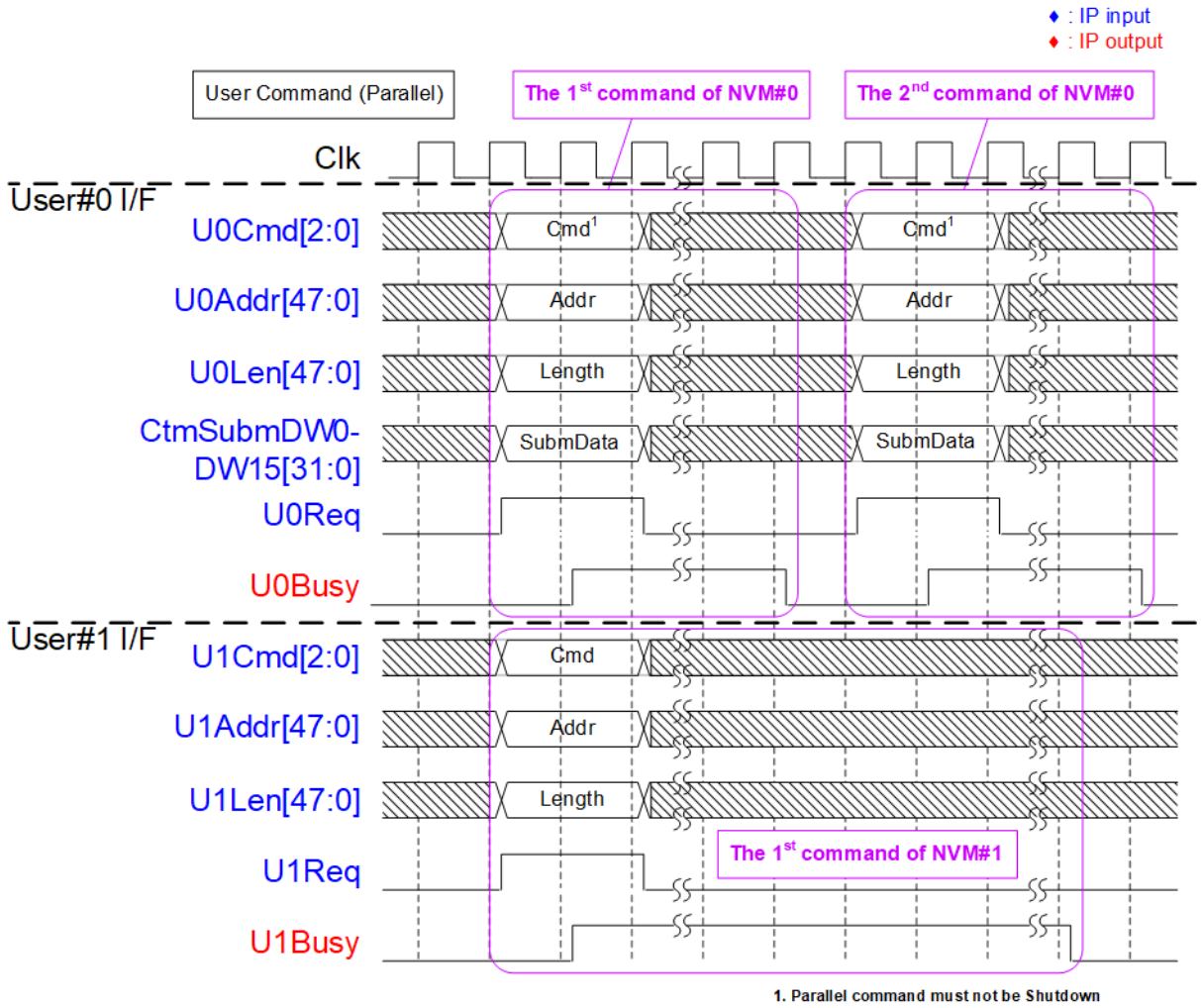

Figure 8: dgIF typeS Control Interface timing diagram when running command parallelly

Figure 7 shows dgIF TypeS Control interfaces of two user interfaces when NVM#0 and NVM#1 operate the command parallelly. Similarly, NVM#2 and NVM#3 can also operate the command parallelly. Each user interface has particular control signals to operate user command. Thus, up to four user commands can be operating at the same time, the first command through User#0 I/F and the second, the third, and the fourth command through User#1 I/F, User#2 I/F and User#3 I/F, respectively. However, Shutdown command is an exemption for this parallel mode.

Data interface of dgIF typeS

Data interface of dgIF typeS is utilized to facilitate the transfer of data streams during the execution of Write and Read command. The interface is compliant with the standard FIFO interface. Figure 9 shows dgIF typeS data interface between Transmit FIFO (of UserLogic module) and Data Buffer (of muNVMe IP) during the Write command process through User#0 I/F. 16-bit FIFO read data counter (U0FifoRdCnt) shows total amount of data stored in the transmit FIFO to be monitored before transferring in a burst. The burst size is 512 bytes or 16 cycles of 256-bit data.

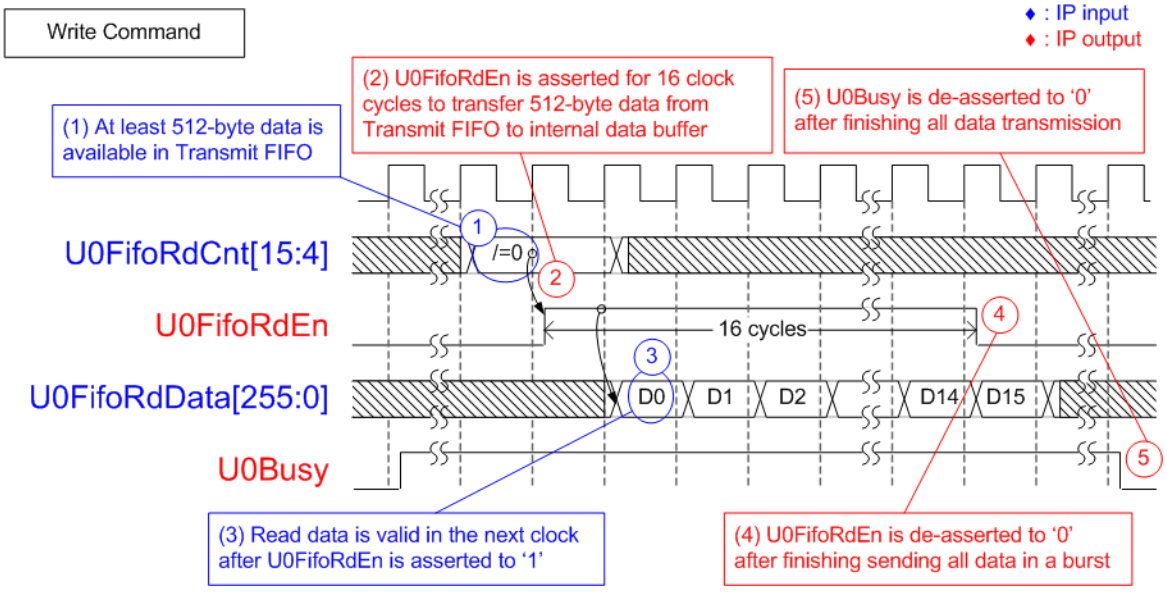

Figure 9: Transmit FIFO interface for NVM#0 Write command

In NVM#0 Write command, data is read from Transmit FIFO#0 until total data are transferred completely. The details to transfer data are described as follows.

1) Before starting a new burst transfer, U0FifoRdCnt[15:4] is monitored. The IP waits until at least 512-byte data is available in Transmit FIFO#0 (U0FifoRdCnt[15:4] is not equal to 0).
2) The IP asserts U0FifoRdEn to '1' for 16 clock cycles to read 512-byte data from Transmit FIFO#0.
3) U0FifoRdData is valid in the next clock cycle after asserting U0FifoRdEn to '1'. 16 data are continuously transferred.
4) U0FifoRdEn is de-asserted to '0' after reading the 16th data (D15). Repeat step 1) – 4) to transfer the next 512-byte data until total amount of data is equal to the transfer size in the command.
5) After total data is completely transferred, U0Busy is de-asserted to '0'.

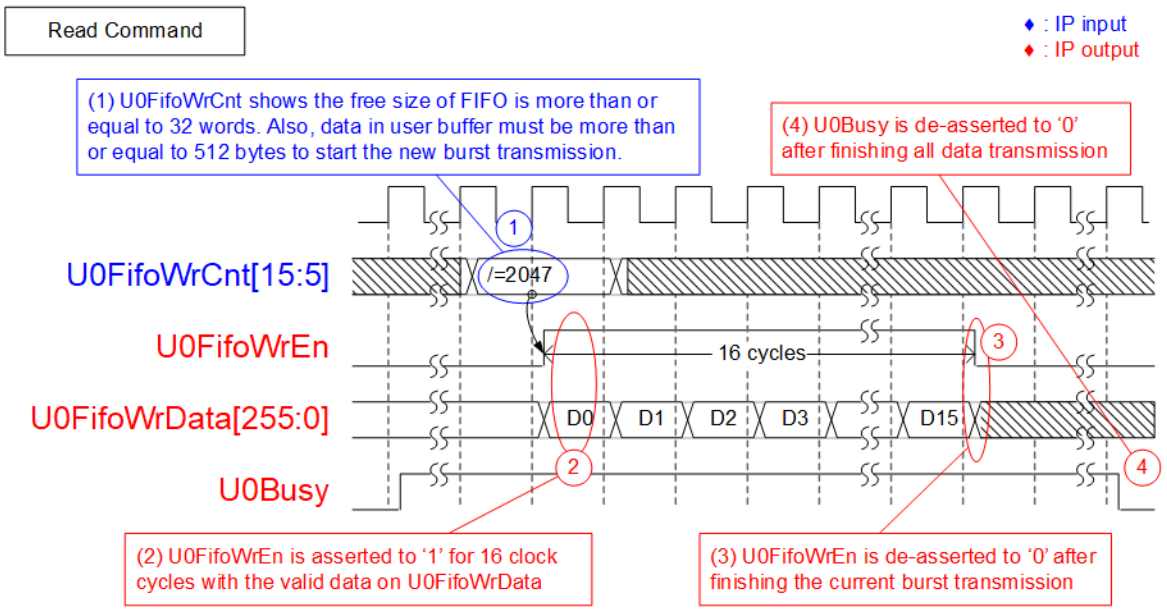

Figure 10: Receive FIFO interface for NVM#0 Read command

In NVM#0 Read command, data is transferred from SSD to Receive FIFO#0 of UserLogic until total data are completely transferred. The details to transfer data are described as follows.

1) Before starting the new burst transmission, U0FifoWrCnt[15:5] is monitored. The IP waits until the free space of Receive FIFO#0 is enough (U0FifoWrCnt[15:5] is not equal to all 1 or 2047). After received data from the SSD is more than or equal to 512 bytes, the new burst transmission begins.
2) The IP asserts U0FifoWrEn to '1' for 16 clock cycles to transfer 512-byte data from the Data Buffer#0 to UserLogic module.
3) After finishing transferring 512-byte data, U0FifoWrEn is de-asserted to '0'. Repeat step 1) – 3) to transfer the next 512-byte data until total data size is equal to the transfer size in the command.
4) After total data is completely transferred, U0Busy is de-asserted to '0'.

Similar to Figure 9 and Figure 10 that show User#0 I/F, dgIF TypeS Data interface of NVM#1-#3 has the same concept by using Transmit FIFO#1-#3, Receive FIFO#1-#3, and Data Buffer#1-#3 to transfer data in NVM#1-#3 Write command and Read command, respectively.

As shown in Figure 8 that the command on User#0 I/F and User#1 I/F are requested parallelly. When parallel command is sent, the parallel data on two user interfaces may be found.

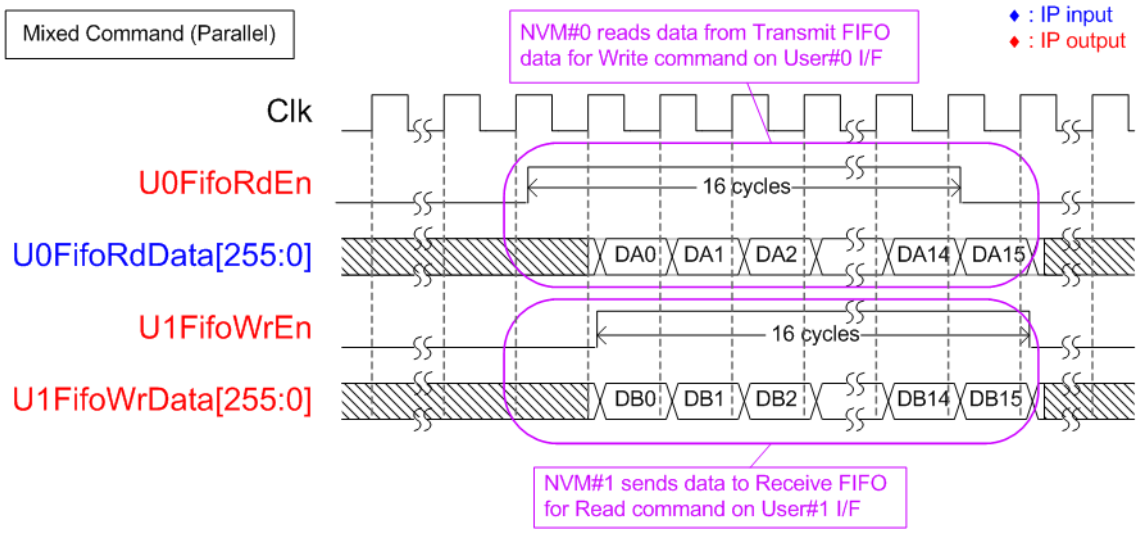

Figure 11: dgIF typeS Data interface timing diagram in parallelly

Figure 11 shows dgIF typeS Data interface of two user interfaces when NVM#0 runs Write command by receiving the Write data and NVMe#1 runs Read command by transmitting the Read data in parallelly. Similarly, NVM#2 and NVM#3 can be operating parallelly by using User#2 Data I/F and User#3 Data I/F that consist of U2-U3FifoRdCnt, U2-U3FifoRdEn, U2-U3FifoRdData, U2-U3FifoWrCnt, U2-U3FifoWrEn, and U2-U3FifoWrData.

IdenCtrl/IdenName

It is recommended to send Identify command to the IP as the first command after the system boots up. This command updates the necessary information of SSD, i.e., total capacity (LBASize) and LBA unit size (LBAMode). The SSD information is applied to be the limitation of the input parameters when operating Write and Read command, described as follows.

1) The sum of the address and transfer length in each user (UXAddr and UXLen; X is the index of the user which can be 0-3) of Write and Read command must not be more than total capacity (LBASize) of the SSD.
2) If LBAMode of the SSD is equal to '1' (LBA unit size is 4 Kbyte), the three lower bit (bit[2:0]) of U0-U3Addr and U0-U3Len must be always equal to '0' to align 4 Kbyte unit.

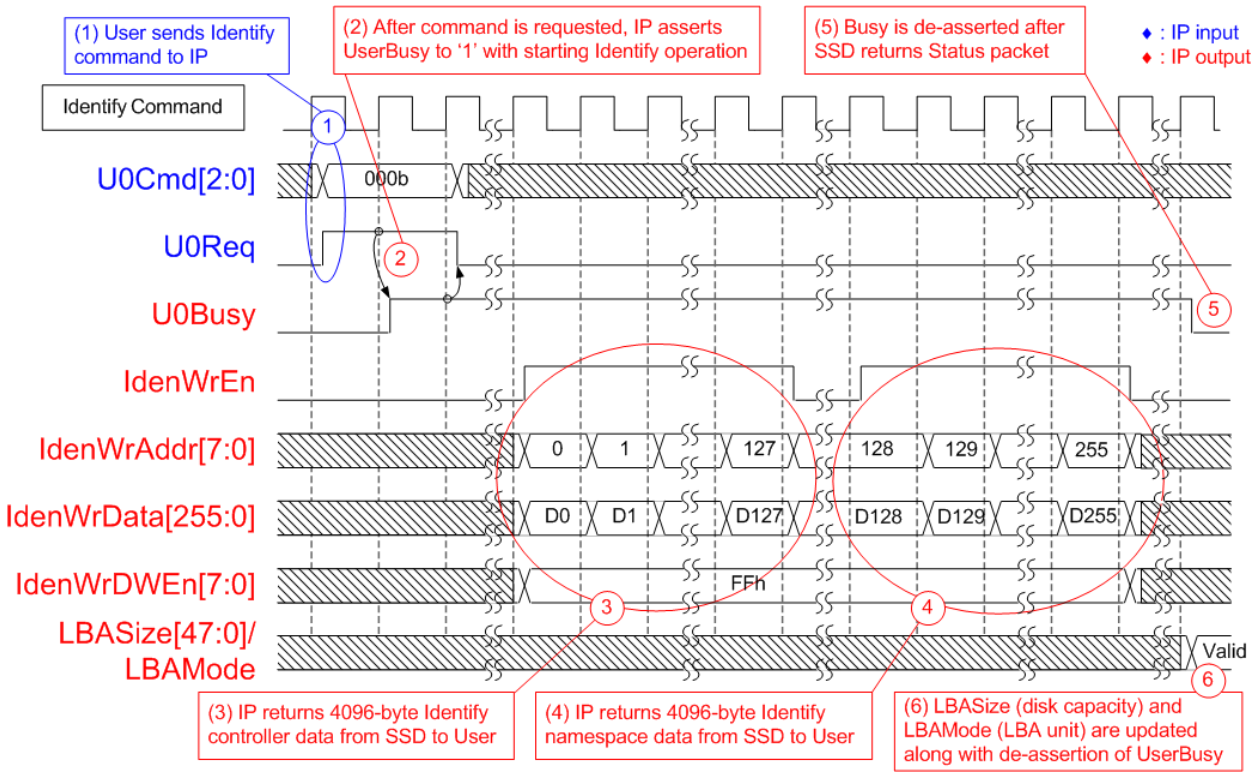

Figure 12: Identify command timing diagram

The details when running Identify command are shown as follows.

1) Send Identify command to the IP (U0Cmd=000b and U0Req='1').
2) The IP asserts U0Busy to '1' after running Identify command.
3) 4096-byte Identify controller data is returned to user. IdenWrAddr is equal to 0-127 with asserting IdenWrEn. Also, IdenWrData and IdenWrDWEn are valid at the same clock as IdenWrEn='1'.
4) 4096-byte Identify namespace data is returned. IdenWrAddr is equal to 128-255. IdenWrAddr[7] can be applied to check data type which is Identify controller data or Identify namespace data.
5) U0Busy is de-asserted to '0' after finishing the Identify command.
6) LBASize and LBAMode of the SSD are simultaneously updated.

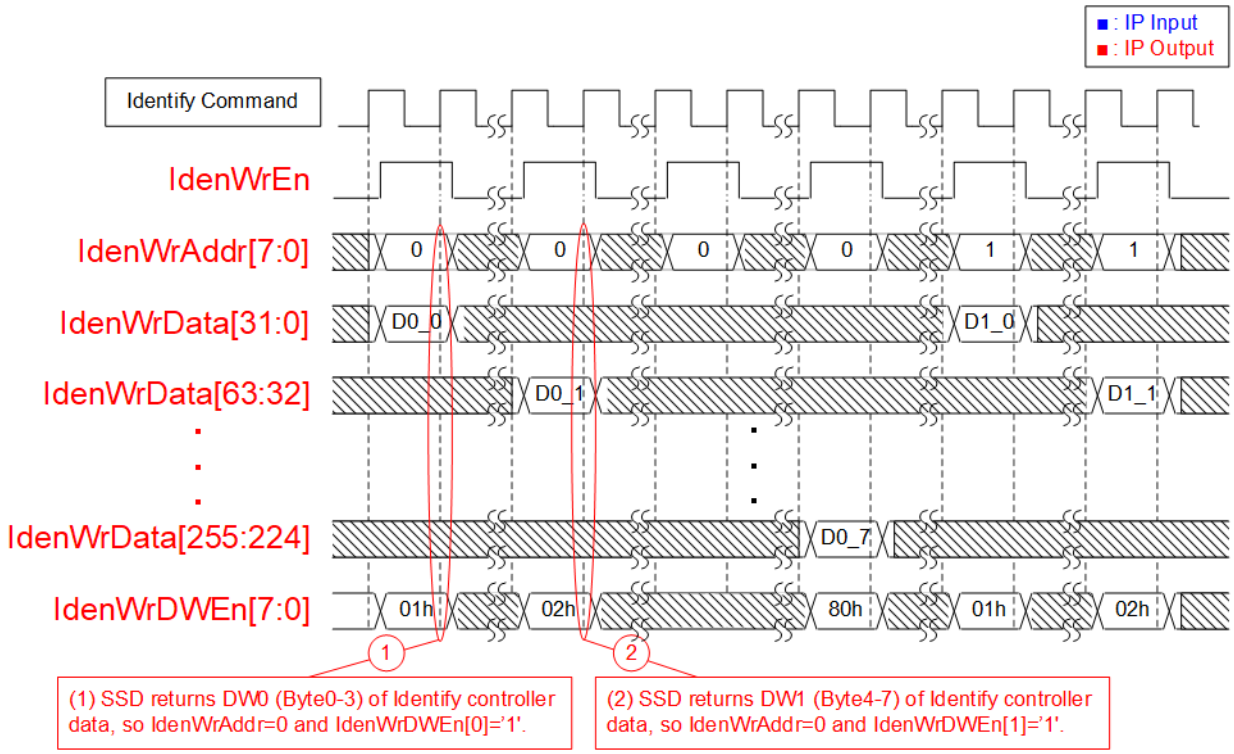

Figure 13: IdenWrDWEn timing diagram

IdenWrDWEn is 8-bit signal to be valid signal of 32-bit data. Some SSDs do not return 4K-byte Identify controller data and Identify namespace data continuously, but it returns only one Dword (32-bit) at a time. Therefore, one bit of IdenWrDWEn is asserted to '1' in the write cycle to write 32-bit data, as shown in Figure 13. IdenWrDWEn[0], [1], …, [7] corresponds to IdenWrData[31:0], [63:32], …, [255:224], respectively.

Shutdown

Shutdown command is recommended to send as the last command before the system is powered down. When Shutdown command is issued, SSD flushes the data from the internal cache to flash memory. After Shutdown command is done, muNVMe IP and SSD are inactive until the system is powered down. If the SSD is powered down without Shutdown command, the total count of unsafe shutdowns (returned data of SMART command) is increased.

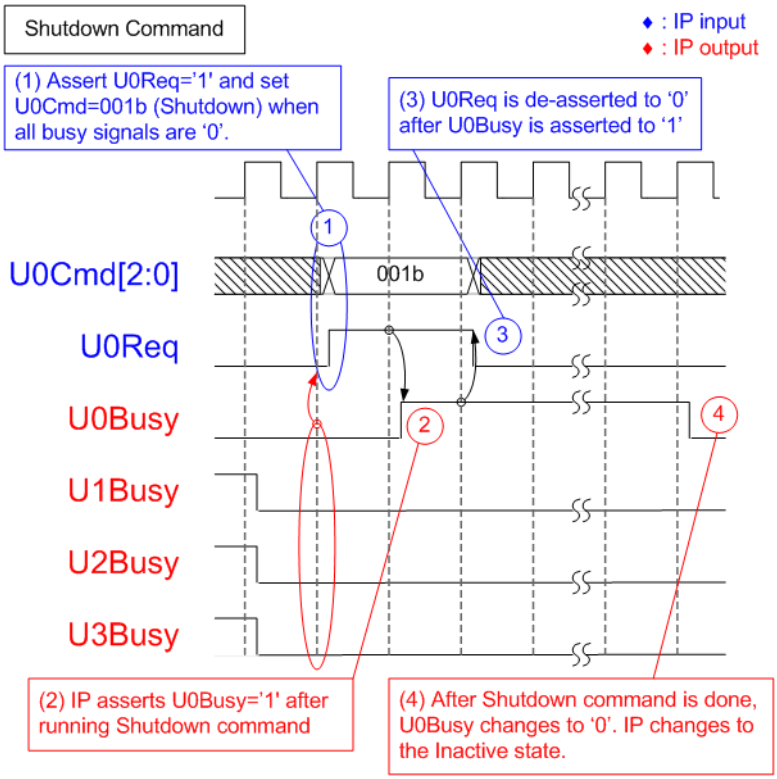

Figure 14: Shutdown command timing diagram

The details when running Shutdown command are shown as follows.

1) Before sending the command request, the IP must be Idle (U0Busy='0'). To send Shutdown command, user asserts U0Req to '1' with U0Cmd=001b.
2) U0Busy is asserted to '1' after muNVMe IP runs Shutdown command.
3) U0Req is de-asserted to '0' to clear the current request after U0Busy is asserted to '1'.
4) U0Busy is de-asserted to '0' when the SSD is shut down completely. After that, the IP does not receive any commands requested from user.

SMART

SMART command is the command to check the SSD health. After sending SMART command, 512-byte health information is returned from the SSD. SMART command loads the parameters from CtmSubmDW0-DW15 signals on Custom command interface. User sets 16-dword data as constant value for SMART command before asserting U0Req. After that, the SMART data is returned via CtmRAM port as shown in Figure 15.

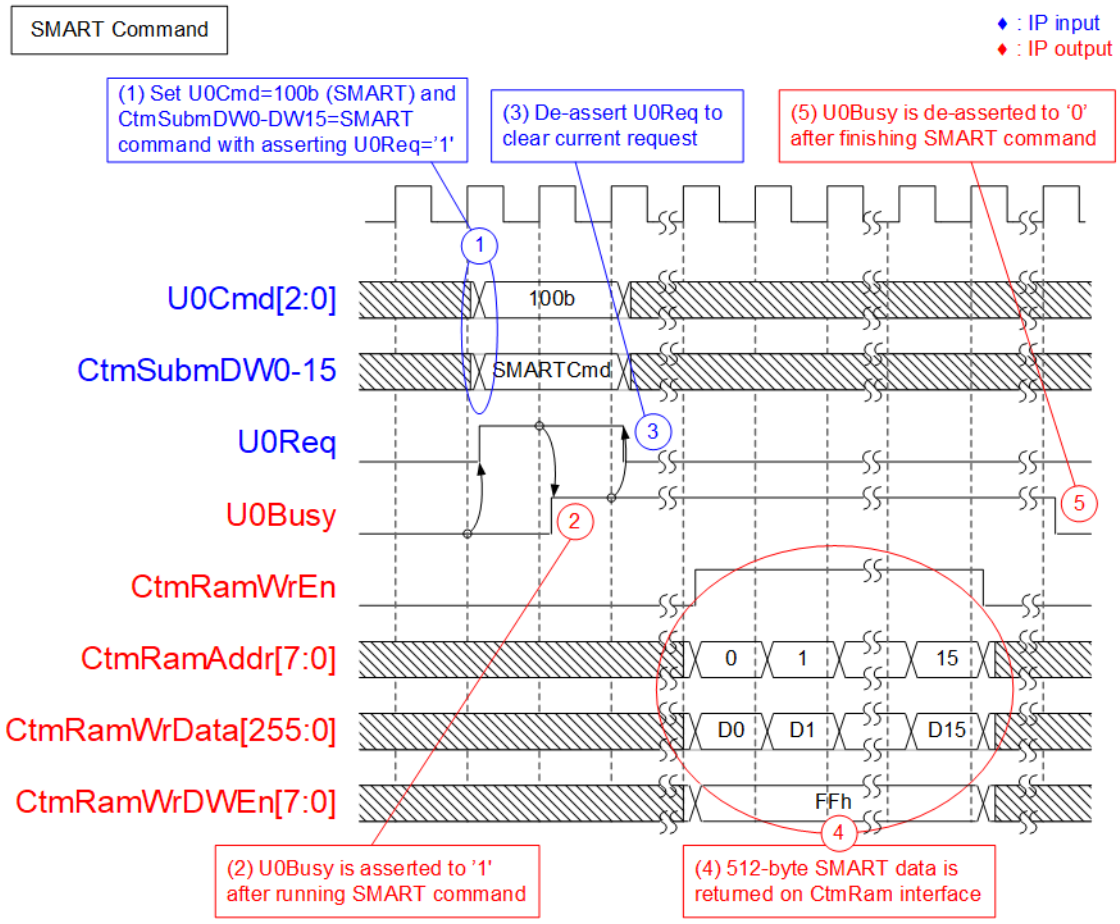

Figure 15: SMART command timing diagram

The details when running SMART command are shown as follows.

1) Before sending the command request, NVM#0 must be Idle (U0Busy='0').
   All input parameters must be stable when U0Req is asserted to '1' for sending the request.
   CtmSubmDW0-DW15 is set as constant value by following value for SMART command.
   CtmSubmDW0 = 0x0000_0002
   CtmSubmDW1 = 0xFFFF_FFFF
   CtmSubmDW2 – CtmSubmDW5 = 0x0000_0000
   CtmSubmDW6 = 0x0802_2000
   CtmSubmDW7 – CtmSubmDW9 = 0x0000_0000
   CtmSubmDW10 = 0x007F_0002
   CtmSubmDW11 – CtmSubmDW15 = 0x0000_0000
2) Assert U0Busy to '1' after muNVMe IP runs SMART command.
3) U0Req is de-asserted to '0' to clear the current request. Next, user logic can change the input parameters for the next command request.
4) 512-byte SMART data is returned on CtmRamWrData signal with asserting CtmRamWrEn to '1'. CtmRamAddr is equal to 0-15 to be data index of 512-byte data. When CtmRamAddr=0, byte0-31 of SMART data is valid on CtmRamWrData. CtmRamWrDWEn is Dword enable for each 32-bit CtmRamWrData. If CtmRamWrDWEn=FFh, all 256 bits of CtmRamWrData are valid.
5) U0Busy is de-asserted to '0' when finishing SMART command.

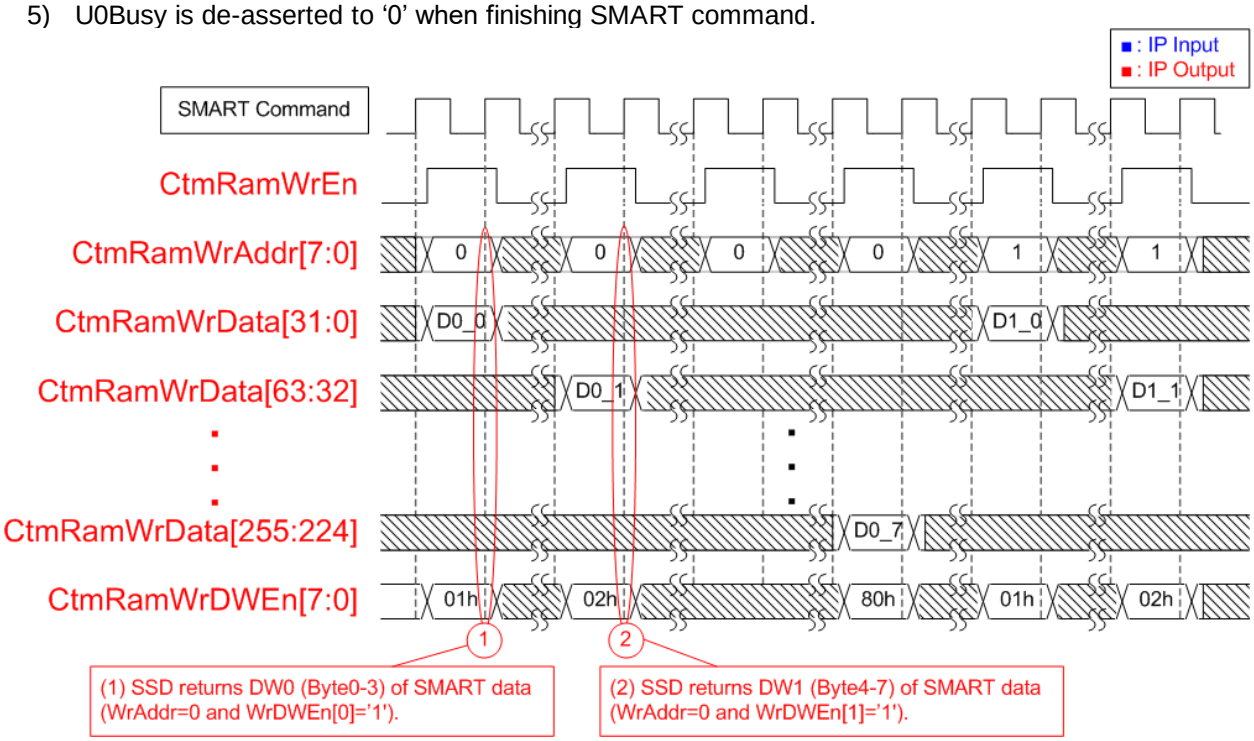

Figure 16: CtmRamWrDWEn timing diagram

Similar to Identify command, some SSDs do not return 512-byte data continuously but return only one Dword (32-bit) at a time. Therefore, one bit of CtmRamWrDWEn is asserted to '1' in the write cycle to be the valid signal of 32-bit CtmRamWrData. CtmRamWrDWEn[0], [1], …, [7] corresponds to CtmRamWrData[31:0], [63:32], …, [255:224], respectively.

Flush

Most SSDs accelerate write performance by storing write data to cache before flushing to the flash memory by the SSD controller. If power is down unexpectedly, the data in the cache may be lost and not stored to the flash memory. Flush command is the command to force the SSD controller to flush data from the cache. After sending Flush command, all data in previous Write command can be guaranteed.

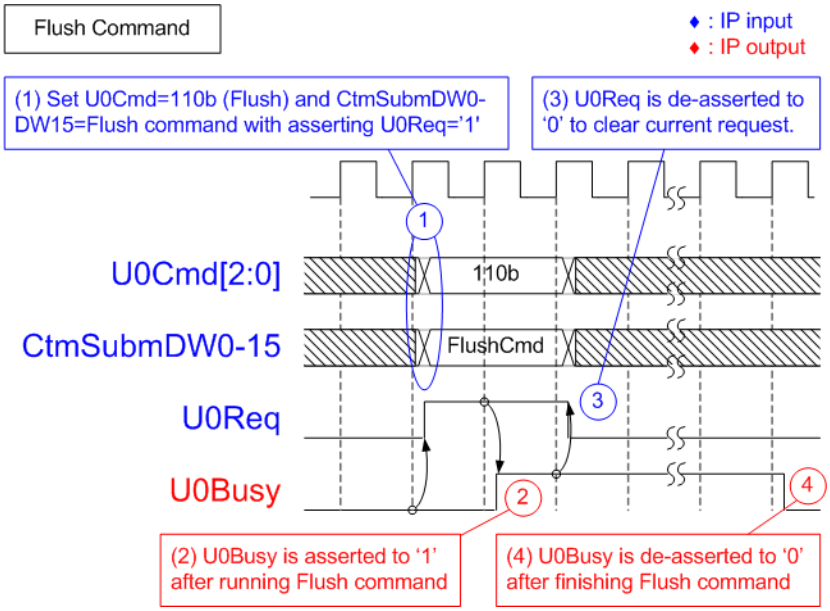

Figure 17: Flush command timing diagram

The details for running Flush command are shown as follows.

1) Before sending the command request, NVM#0 must be Idle (U0Busy='0').
   All input parameters must be stable when U0Req is asserted to '1' for sending the request.
   CtmSubmDW0-DW15 is set as constant value by following value for Flush command.
   CtmSubmDW0 = 0x0000_0000
   CtmSubmDW1 = 0x0000_0001
   CtmSubmDW2 – CtmSubmDW15 = 0x0000_0000
2) U0Busy is asserted to '1' after muNVMe IP runs Flush command.
3) U0Req is de-asserted to '0' to clear the current request. Next, user logic can change the input parameters for the next command request.
4) U0Busy is de-asserted to '0' when Flush command is done.

Error

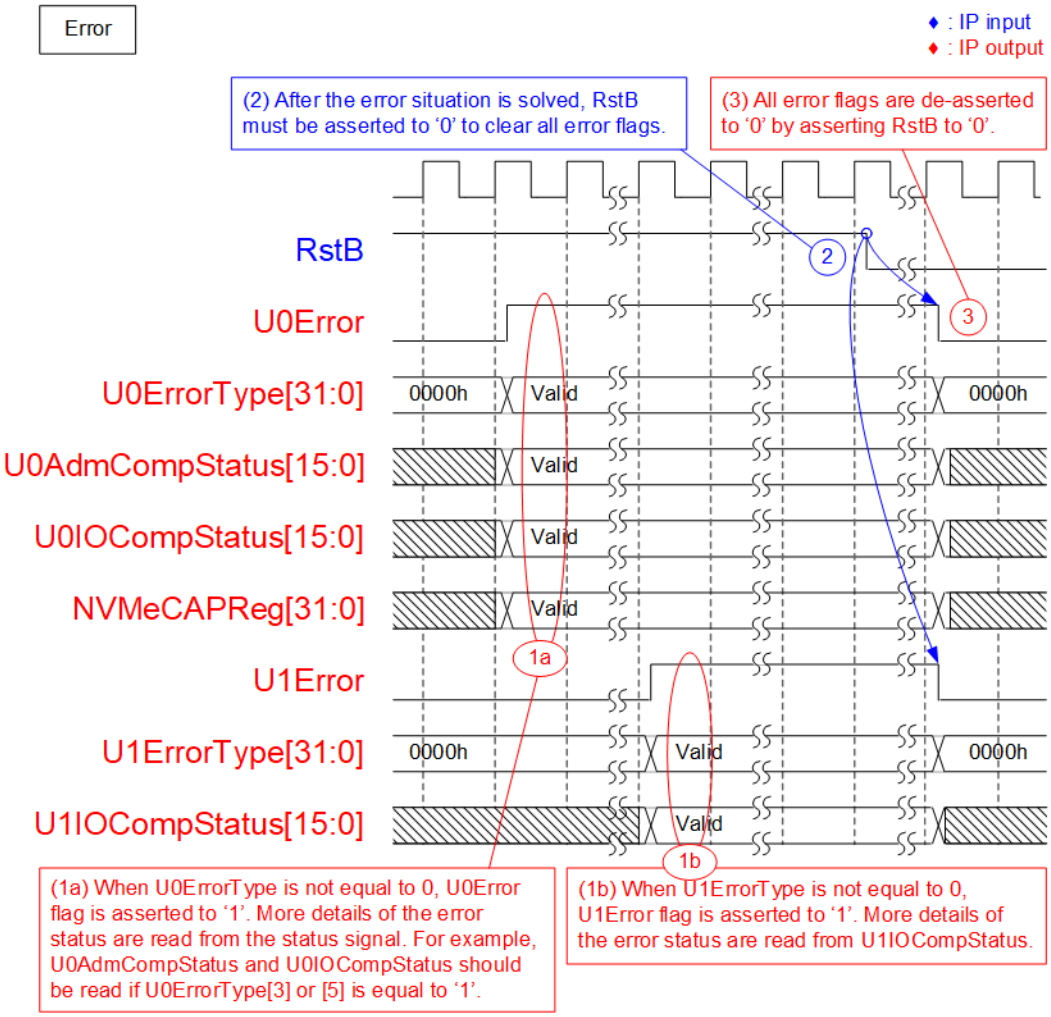

Figure 18: Error flag timing diagram

When the error is found while running initialization process or operating some commands, U0-U3Error flag is asserted to '1'. U0-U3ErrorType is read to check the occurred error type. For User#0 I/F, NVMeCAPReg, U0AdmCompStatus, and U0IOCompStatus are valid for monitoring error details after U0Error is asserted to '1'. Similarly, U1-U3IOCompStatus is valid for monitoring error details on user interface#1-#3 after U1-U3Error is asserted to '1'.

When the error is found while running initialization process, it is recommended to read NVMeCAPReg to check capability of NVMe SSD. When the error is found while operating the command, it is recommended to read U0AdmCompStatus or U0-U3IOCompStatus.

- When bit[3] of U0ErrorType is asserted, read AdmCompStatus to check more details.
- When bit[5] of U0-U3ErrorType is asserted, read U0-U3IOCompStatus to check more details.

U0-U3Error flag are cleared by RstB signal only. After the failure is solved, RstB is asserted to '0' to clear the error flags.

Verification Methods

The muNVMe IP Core for Gen4 functionality was verified by simulation and also proved on real board design by using VCK190 evaluation board.

Recommended Design Experience

Experience design engineers with a knowledge of Vivado Tools should easily integrate this IP into their design.

Ordering Information

This product is available directly from Design Gateway Co., Ltd. Please contact Design Gateway Co., Ltd. For pricing and additional information about this product using the contact information on the front page of this datasheet.

Revision History

Revision	Date	Description
1.0	31-Jan-2023	Initial Release