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 Netlist
Instantiation Templates	VHDL
Reference Designs & Application Notes	Vivado Project, See Reference Design Manual
Additional Items	Demo on KCU105, ZCU106, KCU116, VCU118
Support	
Support Provided by Design Gateway Co., Ltd.	

Features

- NVMe host controller for access one NVMe SSD without CPU and external memory
- Support up to 32 Write or Read commands for 4 Kbyte random access (Multiple mode)
- Supported command (Single mode): Identify, Shutdown, SMART, and Flush
- Not support mixed Write and Read command in the same queue (1 queue = 32 commands)
- Include 128 Kbyte RAM to be data buffer
- Simple user interface by using data stream interface
- 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)
  - MQES (Maximum Queue Entries Supported): More than or equal to 63
  - LBA unit: 512 bytes
- User clock frequency must be more than or equal to PCIe clock (250MHz for Gen3)
- Operating with Integrated Block for PCI Express by using 4-lane PCIe Gen3 (128-bit bus interface)
- Available reference designs: Standard, Data stream design, and Multiple user design on KCU105, ZCU106, KCU116, and VCU118 board with AB18-PCIeX16, AB17-M2FMC, or AB16-PCIeXOVR adapter board
- Customized service for following features
  - Additional NVMe commands
  - RAM size extension to support more than 32 commands or more than 4 Kbyte data size for one Write/Read command

Note: To support mixed Write and Read command, please see more details of our rmNVMe IP.

Table 1: Example Implementation Statistics

Family	Example Device	Fmax (MHz)	CLB Regs	CLB LUTs	CLB	BRAM Tile[1]	URAM	Design Tools
Kintex-Ultrascale	XCKU040FFVA1156-2E	400	3901	2406	732	34	-	Vivado2019.1
Kintex-Ultrascale+	XCKU5P-FFVB676-2E	400	3895	2379	680	34	-	Vivado2019.1
Zynq-Ultrascale+	XCZU7EV-FFVC1156-2E	400	3895	2387	664	34	-	Vivado2019.1
Virtex-Ultrascale+	XCVU9P-FLGA2104-2L	400	3895	2383	668	34	-	Vivado2019.1

Notes: 1) Actual logic resource dependent on percentage of unrelated logic

Applications

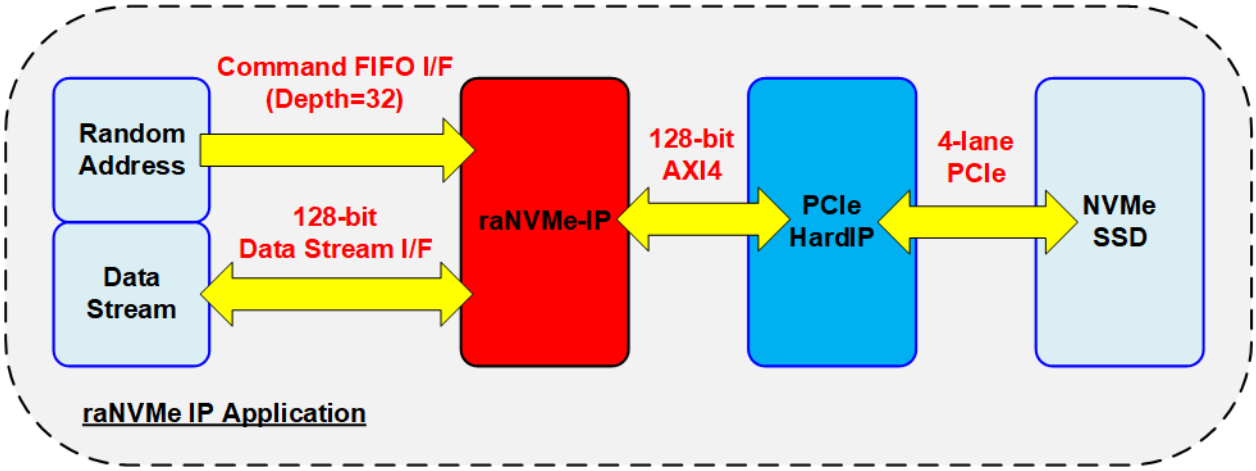

Figure 1: raNVMe IP Application

raNVMe IP Core integrated with Integrated Block for PCI Express (PCIe hard IP) from Xilinx is ideal to access NVMe SSD without CPU and external memory in random access. raNVMe IP with PCIe hard IP can connect to one NVMe SSD and it is recommanded for the application which needs to access NVMe by using multiple addresses with less latency time. raNVMe IP supports to handle up to 32 Write commands or Read commands at the same time. Some applications of raNVMe IP are shown in Figure 2 and Figure 3.

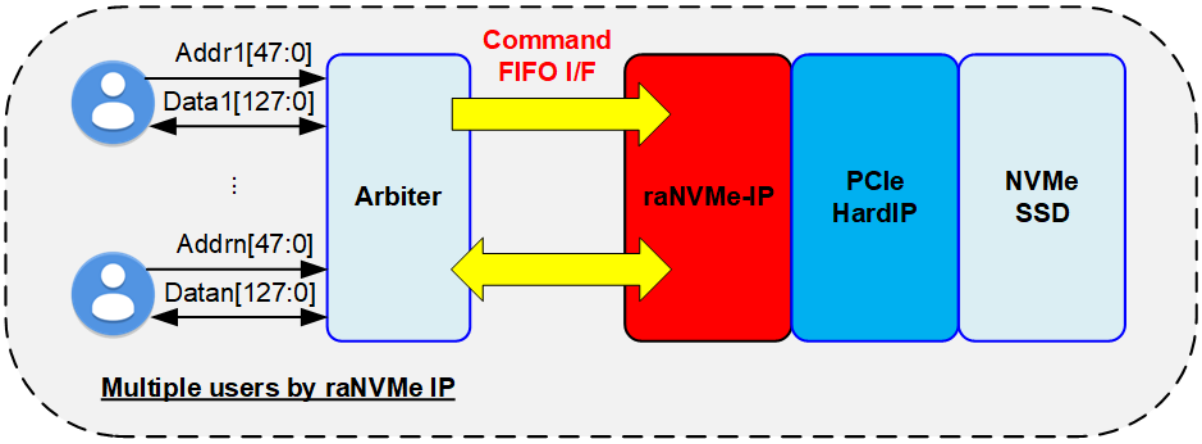

Figure 2: Multiple user application

By designing Arbiter logic connecting between raNVMe-IP and multiple users, many users can transfer data at different addresses of the same NVMe SSD.

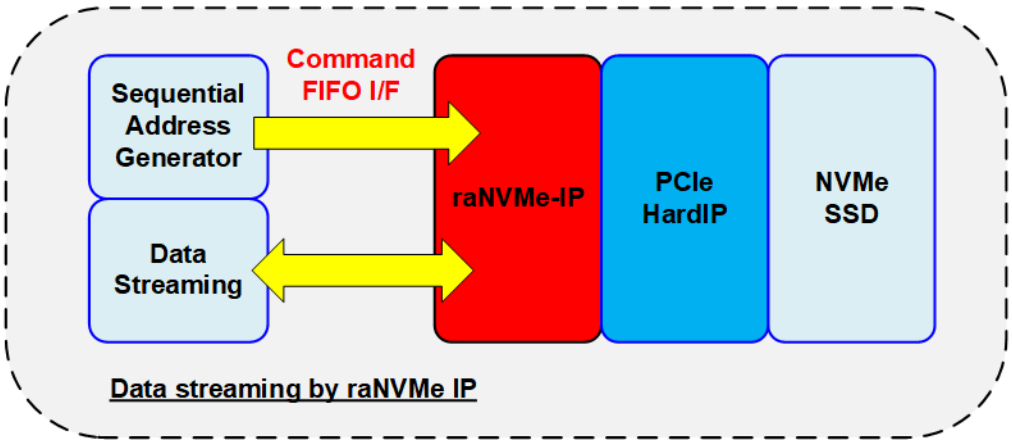

Figure 3: Data streaming application

The last application is data streaming system. Some applications need to record or replay the data from SSD but does not know the total transfer size. The control signals of the application are Start and Stop. One command size of raNVMe IP is fixed to 4 Kbyte and Start/Stop system can be designed by sending multiple 4KB commands in the sequential addressing, as shown in Figure 3.

General Description

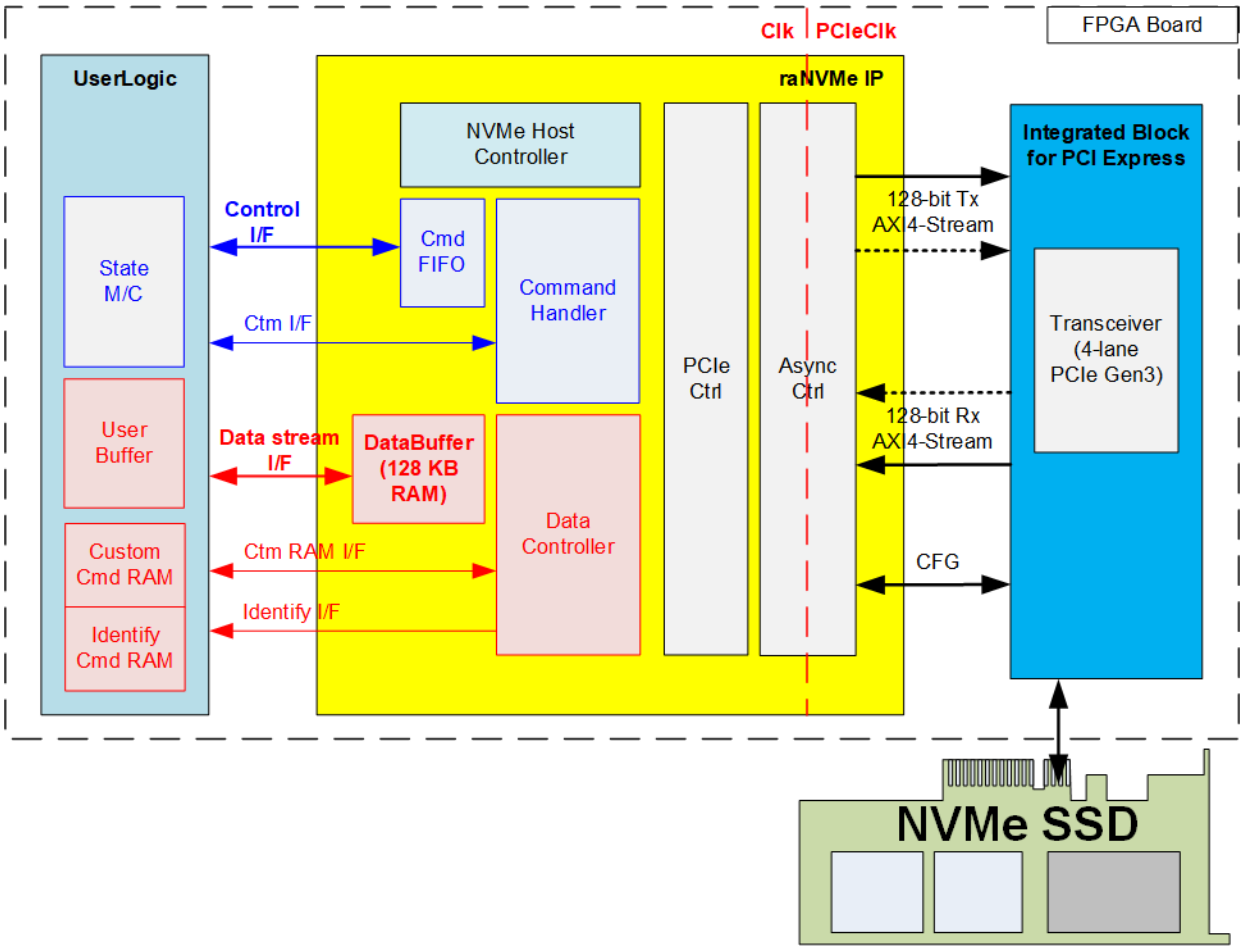

Figure 4: raNVMe IP Block Diagram

raNVMe IP implements as host controller to access NVMe SSD with random access function 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.

raNVMe IP supports six NVMe commands, i.e., Identify, Shutdown, Write, Read, SMART, and Flush command. As shown in Figure 4, Control interface for requesting command with parameter assignments uses FIFO interface to support many Write/Read command requests from the user. However, the data interface of each command uses different interface. The data interface of Write/Read command uses Data stream interface, 128-bit data bus controlled by valid/ready signal. While Data interface of SMART command and Identify command use Ctm RAM I/F and Identify I/F, respectively. Besides, SMART and Flush command require the additional control interface for parameter assignment, called Custom command interface.

raNVMe IP includes asynchronous circuit to allow the user logic running in the individual clock domain, not PCIe clock. However, clock frequency of user logic (Clk) must be more than or equal to PCIe clock (PCIeClk) frequency (250 MHz for PCIe Gen3).

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

Functional Description

To design NVMe host controller supporting random access, raNVMe IP implements two protocols, i.e., NVMe protocol for interfacing with user and PCIe protocol for interfacing with PCIe hard IP. Figure 4 shows the hardware inside raNVMe IP that can be split to two groups – NVMe and PCIe. More details of each module are described as follows.

NVMe

Six commands that raNVMe IP supports are arranged to two groups – Single mode group and Multiple mode group. Single mode group has four commands – Identify, SMART, Flush, and Shutdown. When user sends Single-mode command request to raNVMe IP via Cmd FIFO interface, the command ready is de-asserted to protect the next command request from the user. The command request is re-asserted after the previous command is done. The Multiple command group consists of Write command and Read command. The user can send up to 32 Write commands or Read commands with the address assignment to Cmd FIFO. The order of the data transferring via Data stream I/F is the same as the order of the command that sent to Cmd FIFO (Control I/F). However, all 32 commands that store to Cmd FIFO must be the same command, Write command or Read command. One Write/Read command transfers 4 Kbyte data, so 128-Kbyte buffer is integrated for supporting up to 32 Write/Read commands. The details of each submodule are described as follows.

- **NVMe Host Controller**

  NVMe host controller is the main controller in raNVMe IP. The 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. After finishing the initialization phase, the next phase is operating the command.

  To operate the command, NVMe host controller must handle with Command handler. When the command needs to transfer data such as Write, Read, SMART, and Identify command, NVMe host controller must also handle with Data controller. The transferred order of each packet type received from the SSD is monitored by NVMe host controller. The status value in the received packet is decoded to confirm that the command is finished normally or some errors are found. The error status is returned to the user when some errors are detected by NVMe host controller.

- **Cmd FIFO**

  Command FIFO stores the command and the address which is the parameter of each Write/Read command. Though Cmd FIFO can store up to 32 Write or Read commands, it has the limitation that all commands must be the same command. Cmd FIFO can accept only one command that is Single mode. To change the command, the user must wait until Cmd FIFO is empty.

- **Command Handler**

  This module creates command packet and decodes the status packet returned from the SSD after each command is done. The parameters within the command packet are set by the internal registers when the command is Identify, Shutdown, Write, or Read command. When the command is SMART or Flush command, the parameters are set via 512-bit custom command interface, called Custom submission queue entry which is defined by 16 DWORDs (1 DWORD = 32-bit).

  The status of each command is returned via Completion queue entry. There are three outputs for user monitoring the status extracted from Completion queue entry, i.e., 16-bit Admin completion status when running Identify or Shutdown command, 16-bit IO completion status when running Write or Read command, or 128-bit Custom completion status defined by 4 DWORDs when running SMART or Flush command.

- **Data Buffer**

  128-Kbyte simple dual port RAM is implemented by BlockRAM to be data buffer. The buffer stores data transferring between UserLogic and SSD while Write or Read command is operating.

- **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., Data stream interface with 128-Kbyte buffer when running Write or Read command, Custom command RAM interface when running SMART command, or Identify interface when running Identify command.

  The data controller must allocate the data buffer for transferring the data in each command. When running in Multi-mode command (Write or Read command), the Data controller must handle the order of the data transferred with the user to be the same order as the command input in Cmd FIFO.

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 - PCIeCtrl and AsyncCtrl. More details of each module are described as follows.

- **PCIeCtrl**

  In initialization process, PCIeCtrl sets up PCIe environment of SSD via CFG interface. After that, PCIe packet is created or decoded via 128-bit Tx/Rx AXI4-Stream. The command packet and data packet from NVMe module are converted to be PCIe packet by PCIeCtrl. On the other hand, the received PCIe packet is decoded and converted to NVMe packet for NVMe module by this module.

- **AsyncCtrl**

  AsyncCtrl includes asynchronous registers and buffers to support clock domain crossing. Most logics in raNVMe IP run on user clock domain while PCIe hard IP runs on PCIe clock domain. AXI4-stream interface of PCIe hard IP must transfer data of each packet continuously, so the user bandwidth must be greater than or equal to PCIe bandwidth by running at higher or the same clock frequency of PCIe clock.

User Logic

This module could be designed by using small state machine to send the commands with assigning the parameters for each command. For Write/Read command, user can send commands with the start address to Cmd FIFO. Data interface for Write/Read command is Data stream controlled by valid/ready signal while data output of SMART command and Identify command can be mapped to simple dual port RAM with byte enable. RAM size depends on the data size transferring in each command, but data width of all commands is 128-bit. Data size of Identify command is 8-Kbyte while data size of SMART command is 512-byte.

Integrated Block for PCI Express

Integrated Block for PCI Express is the hard IP provided by Xilinx for some Xilinx FPGAs. The maximum number of SSDs connecting to one FPGA device is limited by the number of PCIe hard IPs in FPGA device. By using raNVMe IP, one PCIe hard IP can conenct to one NVMe SSD. More details of PCIe hard IP are described in following document.

PG156: UltraScale Devices Gen3 Integrated Block for PCI Express

PG213: UltraScale+ Devices Integrated Block for PCI Express

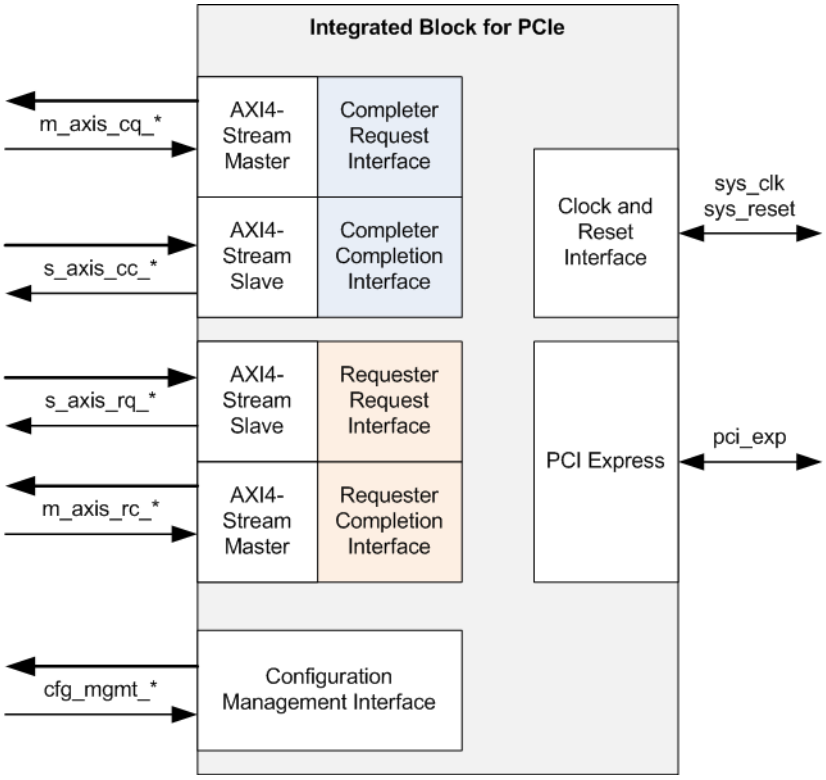

Figure 5: Integrated Block for PCI Express (UltraScale+)

Core I/O Signals

Descriptions of all signal I/O are provided in Table 2 and Table 3.

Table 2: User logic I/O Signals (Synchronous to Clk signal)

Signal	Dir	Description
Control Interface of raNVMe IP		
RstB	In	Synchronous reset signal. Active low. De-assert to '1' when Clk signal is stable.
Clk	In	System clock for running raNVMe IP. The frequency must be more than or equal to PCIeClk which is output from PCIe hard IP (250 MHz for PCIe Gen3).
raNVMCmd[2:0]	In	User command. Valid when raNVMCValid ='1'. (000b: Identify, 001b: Shutdown, 010b: Write SSD, 011b: Read SSD, 100b: SMART, 110b: Flush, 101b/111b: Reserved). *Note: User cannot change to run other commands when the IP is busy. User must wait until the current command is finished (raNVMBusy='0').*
raNVMAddr[47:0]	In	Start address of requested Write/Read command in 512-byte unit. Valid when raNVMCValid ='1'. This value must be less than LBASize which is the SSD capacity. *Note: raNVMAddr[2:0] is ignored for 4-Kbyte alignment.*
raNVMCValid	In	Command valid. Assert to '1' to send new command. When command is Single mode (Identify, Shutdown, SMART, and Flush), raNVMCReady is de-asserted to '0' after the user asserts raNVMeCValid. When command is Multiple mode (Write and Read), user can assert raNVMCValid for sending up to 32 same commands before raNVMCReady is de-asserted to '0'.
raNVMCReady	Out	Asserted to '1' to indicate that raNVMe IP is ready to receive command.
raNVMBusy	Out	IP busy status. Asserted to '1' when raNVMe IP is busy.
raNVMCCnt[5:0]	Out	Remaining command count stored in the Cmd FIFO for running Write/Read command. Valid from 0 to 32. This signal is ignored for Single-mode command.
raNVMCId[4:0]	Out	Command ID in the Cmd FIFO for monitoring in Write/Read command. Valid from 0 to 31 when raNVMCValid='1'. This signal is ignored for Single mode. Used to match with raNVMDId to refer the command number which is currently transferred in Data stream interface. Also, it is used to match with error ID of IOCompStatus[20:16].
LBASize[47:0]	Out	Total capacity of SSD in 512-byte unit which is always aligned to 4 KB unit. This value is valid after Identify command is done. So, Identify command must be the first command that is run to update LBASize value before running other commands. Default value after boot up is 0.
raNVMError	Out	Error flag. Asserted to '1' when raNVMErrorType is not equal to 0. The flag can be cleared by asserting RstB to '0'.
raNVMErrorType[31:0]	Out	Error status. [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 63. 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 equal to 0 or phase tag/command ID is invalid. Please see more details from AdmCompStatus signal. [4] – Error when IO completion entry is not received until timeout.

Signal	Dir	Description
Control Interface of raNVMe IP		
raNVMErrorType[31:0]	Out	[5] – Error when status register in IO completion entry is not equal to 0 or phase tag is invalid. Please see more details from IOCompStatus signal. [6] – Error when Completion TLP packet size is not correct. [7] – Error when PCIe hard IP detects Error correction code (ECC) error from the internal buffer. [8] – Error from Unsupported Request (UR) flag in Completion TLP packet. [9] – Error from Completer Abort (CA) flag in Completion TLP packet. [15:10] – Reserved [16] - Error from unsupport LBA unit (LBA unit is not equal to 512-byte) [31:17] – Reserved *Note: Timeout period of bit[2]/[4] is set byTimeOutSet input.*
Data Stream Interface of raNVMe IP		
raNVMDId[4:0]	Out	Show Command ID (raNVMCId) which is currently transferred on Data stream interface when running Multi-mode commands (Write or Read command). Valid from 0 to 31 when raNVMwValid='1' or raNVMrValid='1'.
raNVMwData[127:0]	In	Write data for Write command. Valid when raNVMwValid='1'.
raNVMwValid	In	Assert to '1' to write data to raNVMe IP for Write command. User can write data to the IP before sending Write command but the IP must be in Idle status. *Note: If other command which is not Write command is run and some write data is stored in the buffer, all write data in the buffer will be flushed.*
raNVMwReady	Out	Asserted to '1' when raNVMe IP accepts the write data (raNVMwData).
raNVMwCnt[13:0]	Out	Total data count of write data in the buffer in 128-bit unit. Valid when running Write command or the IP is in Idle status. When running other commands, this signal is not valid. Also, this signal is reset to 0 after running other commands.
raNVMrData[127:0]	Out	Read data from SSD in Read command. Valid when raNVMrValid ='1'.
raNVMrValid	Out	Read data valid signal. Asserted to '1' to transfer read data. The user can pause data transmission by asserting raNVMrPause to '1'. raNVMrValid is de-asserted to '0' within 4 clock cycles after asserting raNVMrPause to '1'.
raNVMrPause	In	Asserted to '1' to pause received data (raNVMrData) transmission by de-asserting raNVMrValid to '0' within 4 clock cycles.

Signal	Dir	Description
raNVMe IP Interface		
IPVesion[31:0]	Out	IP version number
TestPin[31:0]	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 is disabled.
AdmCompStatus[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
IOCompStatus[31:0]	Out	Status output from IO Completion Entry [0] – Set to '1' when Phase tag in IO Completion Entry is invalid. [15:1] – Status field value of IO Completion Entry [20:16] – Command ID of IO Completion Entry, referred to raNVMCId [31:21] – Reserved
NVMeCAPReg[31:0]	Out	The parameter value of the NVMe capability register when raNVMErrorType[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
IdenWrEn	Out	Asserted to '1' for sending data output from Identify command.
IdenWrDWEn[3:0]	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], and [3] correspond to IdenWrData[31:0], [63:32], [95:64], and [127:96], respectively.
IdenWrAddr[8:0]	Out	Index of IdenWrData in 128-bit unit. Valid when IdenWrEn='1'. Bit[8]='0': 0x000-0x0FF is 4Kbyte Identify controller data. Bit[8]='1': 0x100-0x1FF is 4Kbyte Identify namespace data.
IdenWrData[127:0]	Out	4Kbyte Identify controller data or Identify namespace data. Valid when IdenWrEn='1'.
Custom interface		
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. Valid when raNVMCValid='1' and raNVMCmd=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[3:0]	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], [2], and [3] correspond to CtmRamWrData[31:0], [63:32], [95:64], and [127:96], respectively.
CtmRamAddr[8:0]	Out	Index of CtmRamWrData when SMART data is received. Valid when CtmRamWrEn='1'. (Optional) Index to request data input from CtmRamRdData for customized Custom commands.
CtmRamWrData[127:0]	Out	512-byte data output from SMART command. Valid when CtmRamWrEn='1'.
CtmRamRdData[127:0]	In	(Optional) Data input for customized Custom commands.

Table 3: Physical I/O Signals for PCIe Gen3 Hard IP (Synchronous to PCIeClk)

Signal	Dir	Description
PCIe Gen3 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 Gen3).
PCIeLinkup	In	Assert to '1' when PCIe hard IP is linked up.
Configuration Management Interface		
PCIeCfgDone	In	Read/Write operation complete. Asserted for 1 cycle when the operation completes.
PCIeCfgRdEn	Out	Read enable. Asserted to '1' for a read operation.
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[18:0]	Out	Read/Write Address.
Requester Request Interface		
PCIeMtTxData[127:0]	Out	Requester request data bus.
PCIeMtTxKeep[3:0]	Out	Bit [i] indicates that Dword [i] of PCIeMtTxData contains valid data.
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[59:0]	Out	Requester request user data. Valid when PCIeMtTxValid is high.
PCIeMtTxValid	Out	Asserted to drive valid data on PCIeMtTxData bus. raNVMe IP keeps the valid signal asserted during the transfer of packet.
Completer Request Interface		
PCIeMtRxData[127:0]	In	Received data from PCIe hard IP.
PCIeMtRxKeep[3:0]	In	Bit [i] indicates that Dword [i] of PCIeMtRxData contains valid data.
PCIeMtRxLast	In	Assert this signal in the last beat of a packet to indicate the end of the packet.
PCIeMtRxReady	Out	Indicates that raNVMe 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[127:0]	Out	Completion data from raNVMe IP.
PCIeSlTxKeep[3:0]	Out	Bit [i] indicates that Dword [i] of PCIeSlTxData contains valid data.
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. raNVMe IP keeps the valid signal asserted during the transfer of a packet.
Requester Completion Interface		
PCIeSlRxData[127:0]	In	Received data from PCIe hard IP.
PCIeSlRxKeep[3:0]	In	Bit [i] indicates that Dword [i] of PCIeSlRxData contains valid data.
PCIeSlRxLast	In	Assert this signal in the last beat of a packet to indicate the end of the packet.
PCIeSlRxReady	Out	Indicates that raNVMe IP is ready to accept data.
PCIeSlRxUser[84:0]	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

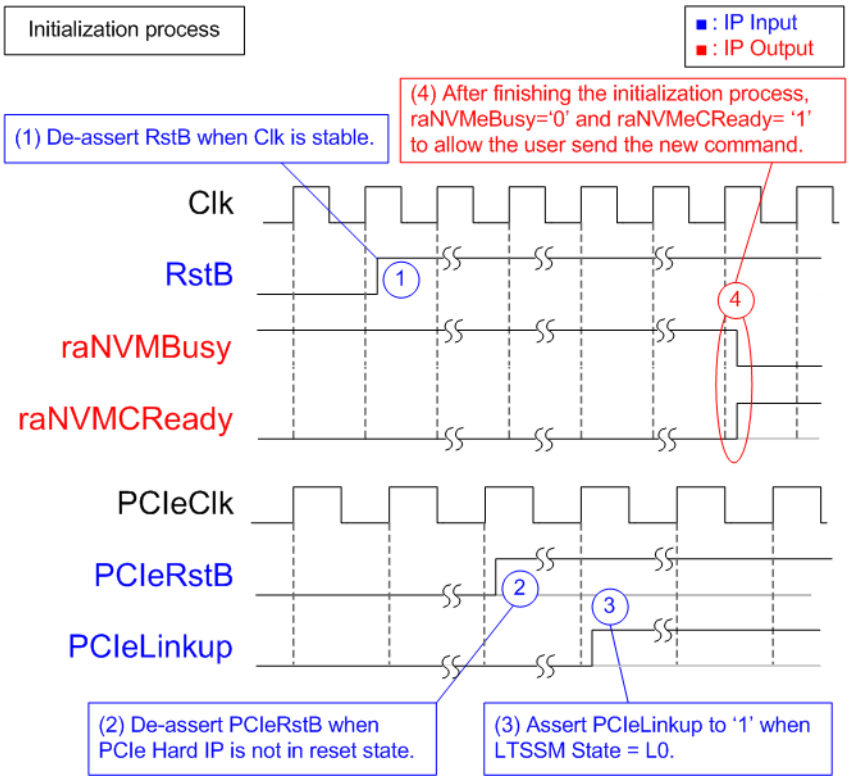

Figure 6: Timing diagram during initialization process

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, raNVMe IP starts initialization process.
4) After finishing the initialization process, raNVMe IP de-asserts raNVMBusy to '0' and asserts raNVMCReady to '1' for receiving the new command from user.

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

Control interface (Single mode)

raNVMe IP supports two command types - Single mode (run one command at a time) and Multiple mode (supports up to 32 commands). Write command and Read command are Multiple mode while other commands are Single mode. raNVMCReady is the status signal to show if the IP is ready to receive the new command or not. The details of Control interface for sending the command are described as follows.

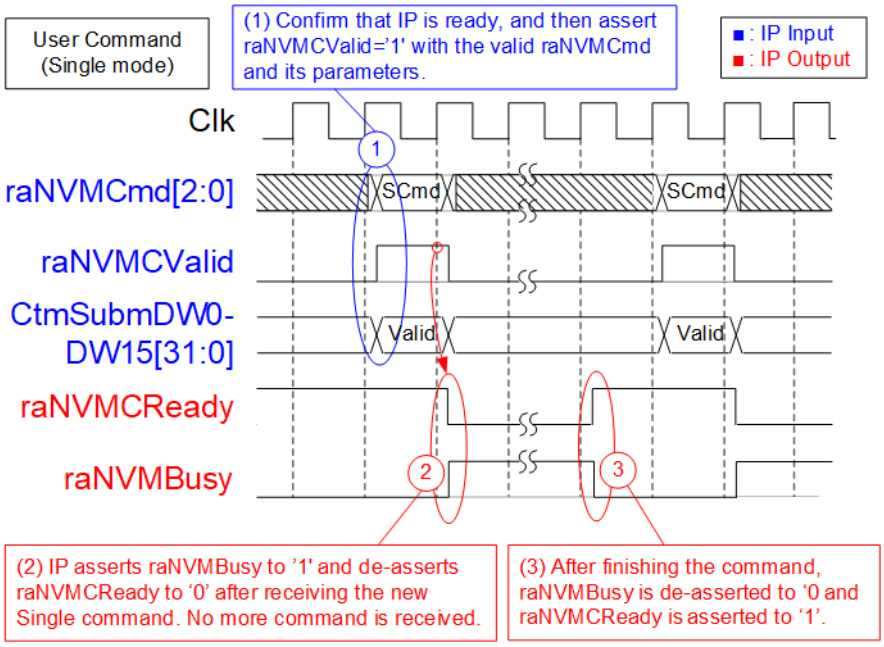

Figure 7: Single command timing diagram

Figure 7 shows timing diagram when running Single command, i.e., Identify, Shutdown, SMART, and Flush command.

1) Before sending the new command to the IP, user must check raNVMeBusy='0' and raNVMCReady='1' to confirm that raNVMe IP is Idle. After that, assert raNVMCValid to '1' for one clock cycle along with valid raNVMCmd and the parameters which depend on the command.
   Note: For SMART and Flush command, the parameters are CtmSubmDW0-DW15.
   For Identify and Shutdown command, no parameter is used.
2) raNVMe IP asserts raNVMBusy to '1' after receiving the new command request and starting the command operation. Also, raNVMCReady is de-asserted to '0' when the receive command is Single mode. After that, user must not send more command requests.
3) After finishing command operation, raNVMBusy is de-asserted to '0' and raNVMCReady is asserted to '1' to accept the new command, except Shutdown command. If Shutdown command is done, raNVMBusy is de-asserted to '0' and raNVMCReady is de-asserted to '0' to wait system shut down without getting more command requests from user.

Control interface (Multiple-mode command)

For Write or Read command which are Multi-mode command, user can send many commands to raNVMe IP until Cmd FIFO is full (32 commands). However, user cannot switch the command from Write to Read and vice versa while the IP is operating (raNVMBusy='1'). To switch the command, the user must wait until all the commands are completed operated by monitoring raNVMBusy='0'.

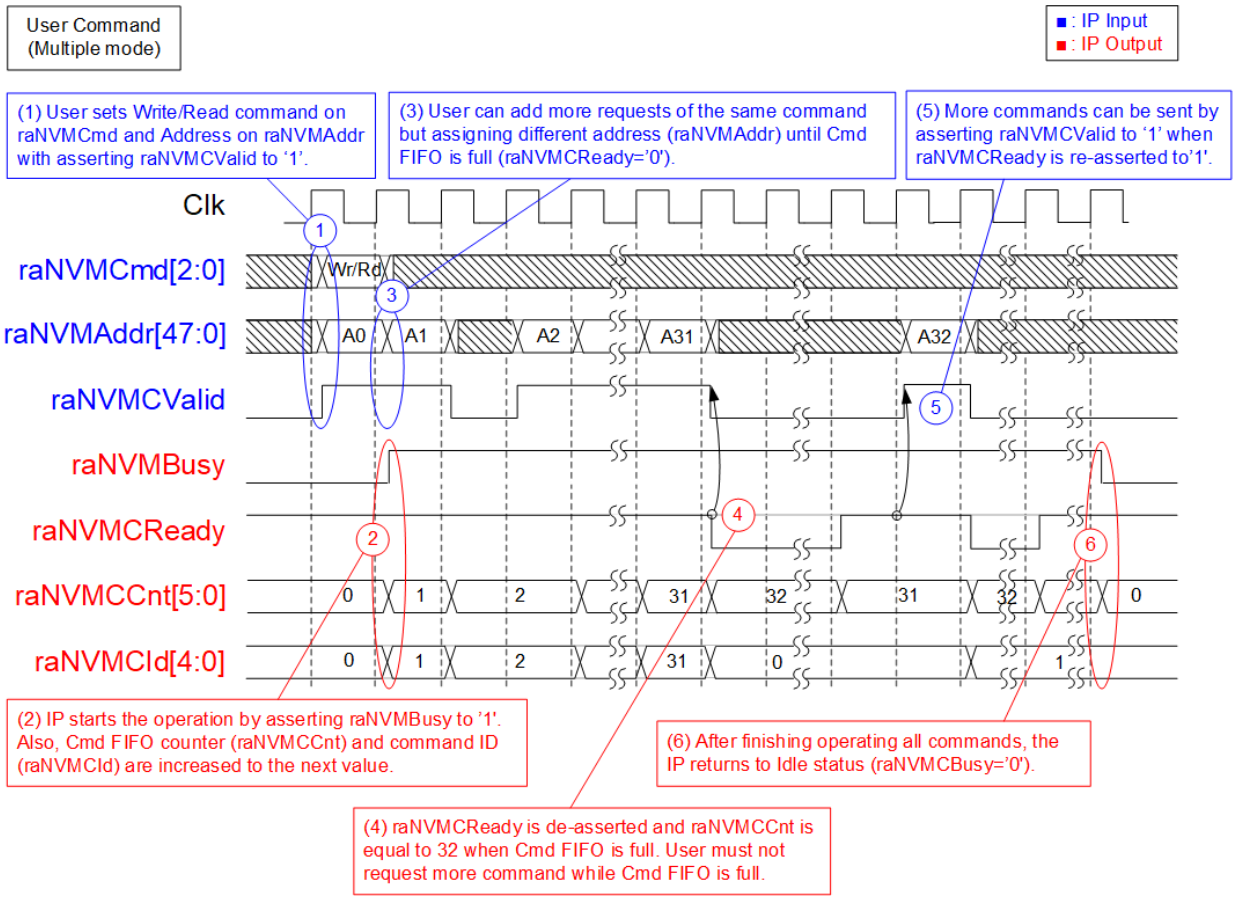

Figure 8: Multi-mode command timing diagram

1) User must comfirm that raNVMe IP is Idle (raNVMeBusy=’0’) and Cmd FIFO is ready (raNVMCReady=’1’). Next, send Write (raNVMCmd=010b) or Read command (raNVMCmd=011b) to the IP by asserting raNVMCValid to ‘1’ for one clock cycle along with the valid raNVMAddr.
2) IP starts the command operation with asserting raNVMBusy to ‘1’. Meanwhile, raNVMCCnt and raNVMCId are increased to the next value for indicating the number of commands in the Cmd FIFO and the next command ID, respectively.
3) As long as Cmd FIFO is not full (raNVMCCnt is less than 32 or raNVMCReady=’1’), user can request of the same command with the new address. raNVMCmd is ignored when the new request is received but raNVMBusy is asserted to ’1’. Therefore, the wrong operation will be happened if user assigns different value of raNVMCmd while raNVMBusy=’1’.
4) After Cmd FIFO is full (raNVMCCnt = 32), raNVMCReady is de-asserted to ‘0’. User shall not request any command. It needs to wait until raNVMCReady is re-asserted again.
5) After some commands are done, Cmd FIFO is not full. User can request the same command to the IP.
6) When raNVMe IP finishes all of the command request, raNVMCBusy is de-asserted to ‘0’. raNVMe IP returns back to the Idle state. It is ready to receive the new command which is different from the previous command.

Data stream interface (Write command)

Data stream interface of raNVMe IP is applied for transferring data of Write command or Read command. The control signals of Data stream interface are valid and ready signal. Figure 9 shows the example of data stream interface when running Write command.

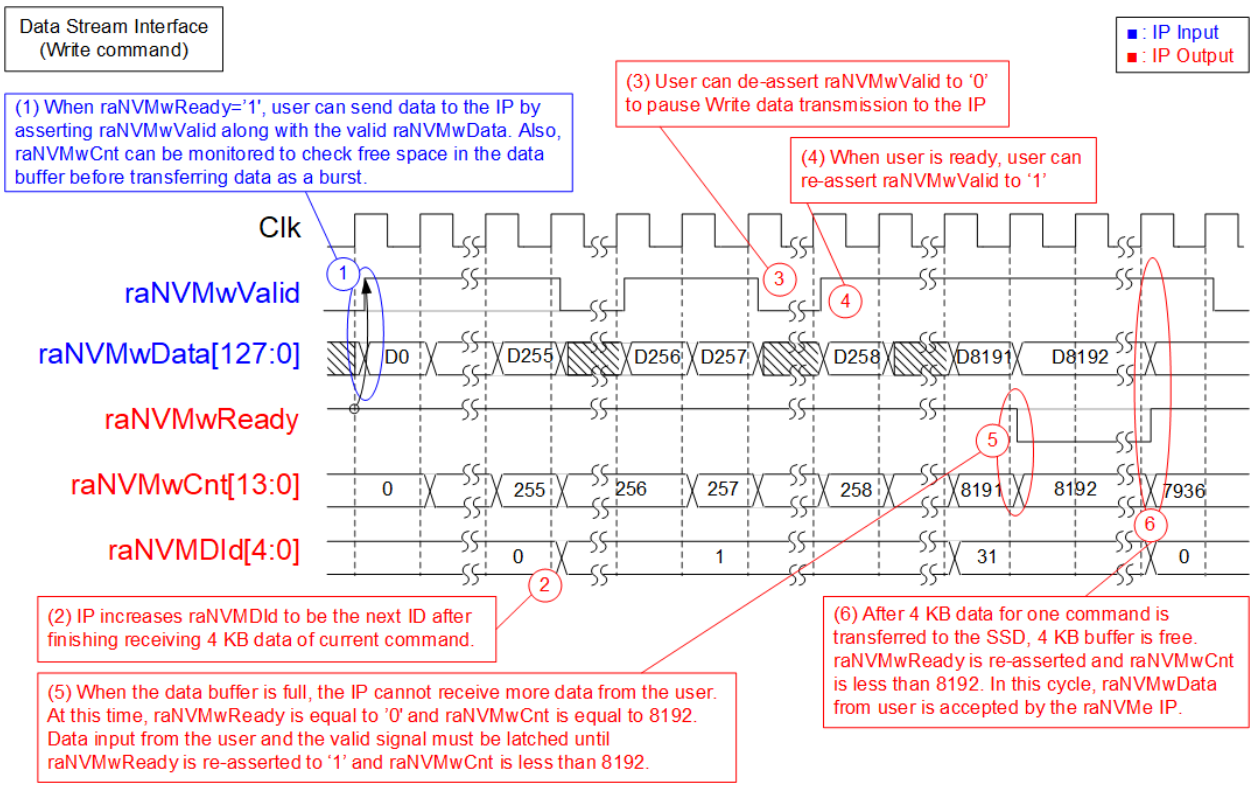

Figure 9: Data stream interface in Write command

In Write command, raNVMe IP Data stream interface operates independently from Control interface. User can prepare the data into the data buffer before or after sending Write command request. The details of data stream interface for the Write command are as follows.

1) User must confirm that the IP is ready to receive data by monitoring raNVMwReady='1' or raNVMwCnt less than 8192. After that, user asserts raNVMwValid along with valid raNVMwData to transfer 128-bit data. There are two status signals to be flow control signals - raNVMwReady and raNVMwCnt. raNVMwCnt is recommended for transferring data as burst mode while raNVMwReady is applied for non-burst data transferring.

   Note: If the user writes the data to raNVMe IP and then runs other command that is not Write command, all Write data will be cleared from the buffer inside raNVMe IP.

2) After transferring each 4 KByte data (data size for one command), raNVMDId is increased to show the command ID of the next data which will be transferred on the Data stream interface. raNVMDId shows the ID is currently transferred on raNVMwData.
3) When user is not ready to send data to IP, user can de-assert raNVMwValid to '0' to pause transferring data.
4) When user is ready to send data to IP, user can re-assert raNVMwValid to '1' to transfer data.
5) When the data buffer is full (raNVMwCnt is equal to 8192 and raNVMwReady is de-asserted to '0'), the IP cannot accept the data from the user. Therefore, raNVMwData and raNVMwValid must hold the same value until raNVMwReady is re-asserted to '1'.
6) After each Write command is done by the SSD, data buffer gets 4 KB free space. Therefore, raNVMwCnt is decreased by 4 KB and raNVMwReady is re-asserted to '1'. The data for the next command can be accepted by the IP.

Data stream interface (Read command)

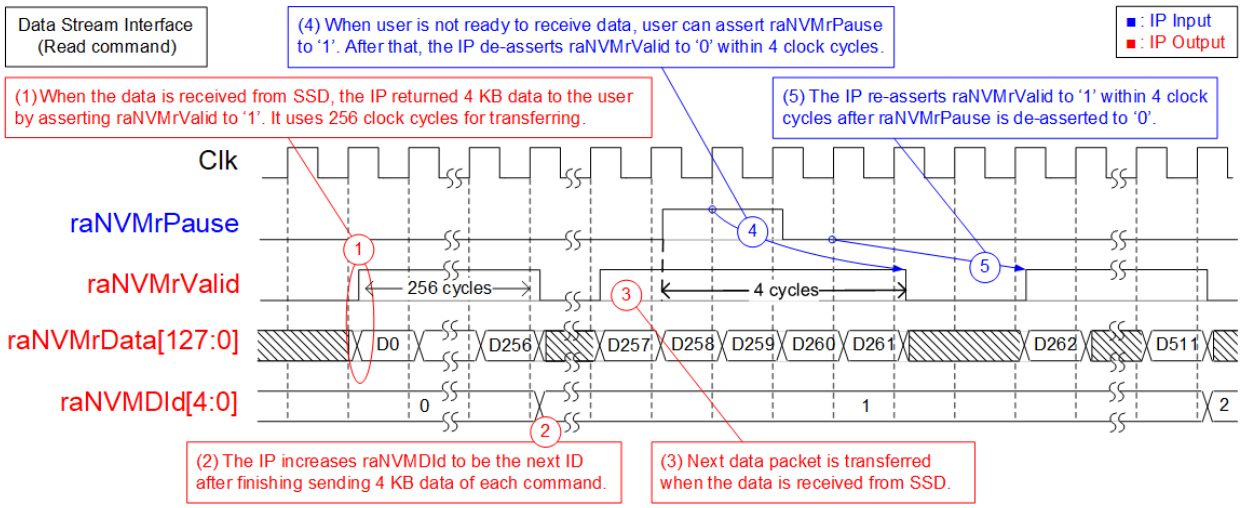

Figure 10: Data stream interface in Read command

Similar to Write command, the Data stream interface of the IP is run independently from the Control interface. The control signals of Read data stream are valid and pause signal. The details of data stream interface for the Read command are as follows.

1) After raNVMe IP receives the 4 KB data of the current command ID from SSD, the IP returns the data to the user by asserting raNVMrValid to '1' along with valid raNVMrData. To transfer 4 KB data, raNVMrValid is asserted to '1' for 256 clock cycles continuously if raNVMrPause is always de-asserted to '0'.
2) After finishing transferring each 4 KB data, raNVMDId is increased to indicate the next command ID that will be transferred.
3) The next 4 KB data is transferred after the SSD returns the 4KB data, similar to step 1).
4) When user is not ready to receive read data, user can assert raNVMrPause to '1'. After that, raNVMe IP pauses data transferring by de-asserting raNVMrValid to '0' within 4 clock cycles.
5) When user is ready to receive read data by de-asserting raNVMrPause to '0', raNVMe IP re-asserts raNVMrValid to '1' within 4 clock cycles to continue to transfer the data.

IdenCtrl/IdenName

It is recommened to send Identify command to the IP as the first command after system boots up. This command updates total capacity (LBASize) which is the necessary information of SSD for calculating the valid range of the address (UserAddr) in Write or Read command. UserAddr in Write and Read command must be less than LBASize of the SSD.

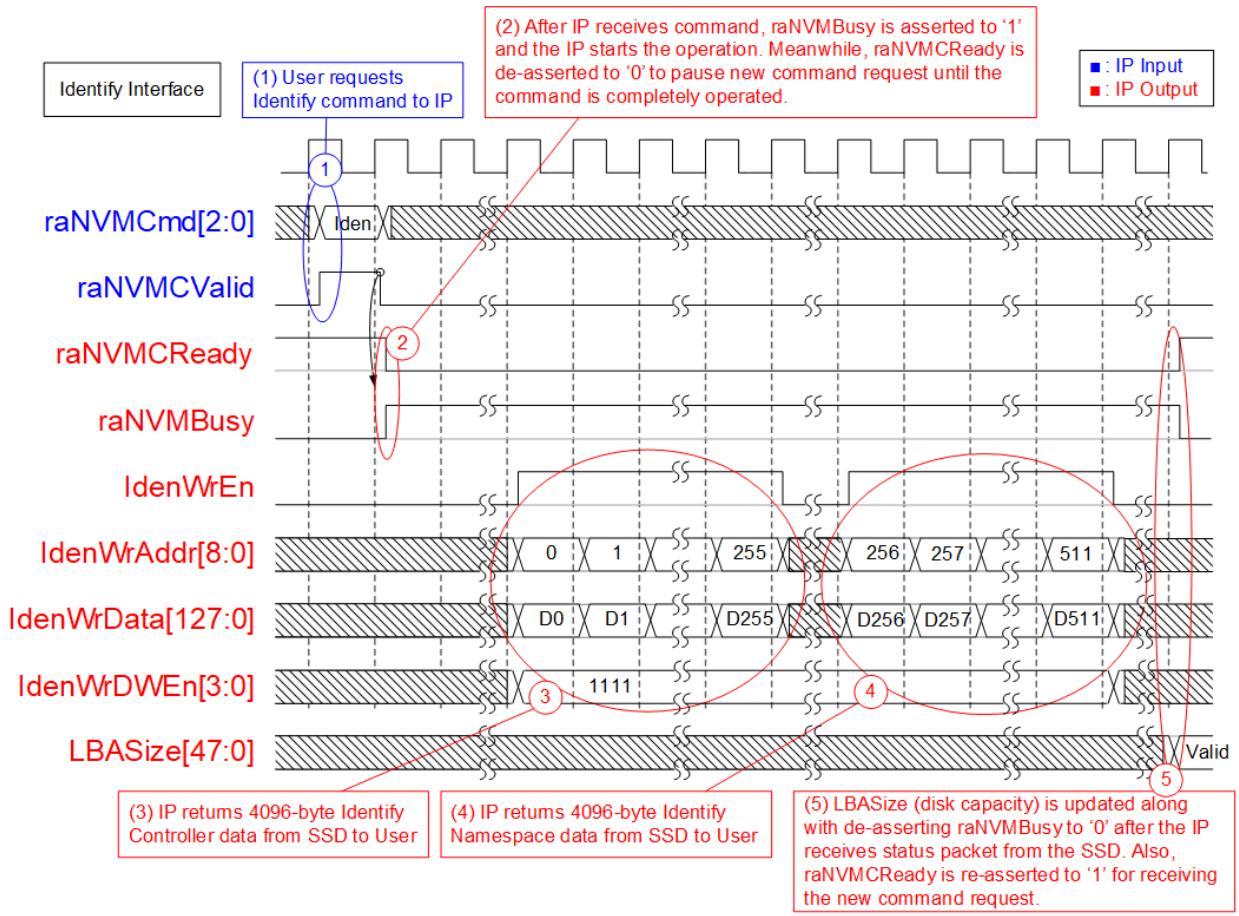

Figure 11: Identify command timing diagram

The details when running Identify command are shown as follows.

1) Send the request of Identify command to the IP (raNVMCmd=000b and raNVMCValid='1').
2) The IP asserts raNVMBusy to '1' after running the command. Meanwhile, raNVMCReady is de-asserted to '0' to ignore the new command request from the user until the current command is completed.
3) 4096-byte Identify controller data is returned to user. IdenWrAddr is equal to 0-255 with asserting IdenWrEn. IdenWrData and IdenWrDWEn are valid at the same clock as IdenWrEn='1'.
4) Similar to Identfy controller data, 4096-byte Identify namespace data is returned but IdenWrAddr is equal to 256-511. Therefore, IdenWrAddr[8] can be applied to check the data type which is Identify controller data or Identify namespace data.
5) After finishing the Identify command operation, raNVMBusy is de-asserted to '0' along with valid LBASize. Also, raNVMCReady is re-asserted to '1' to receive the new command from the user.

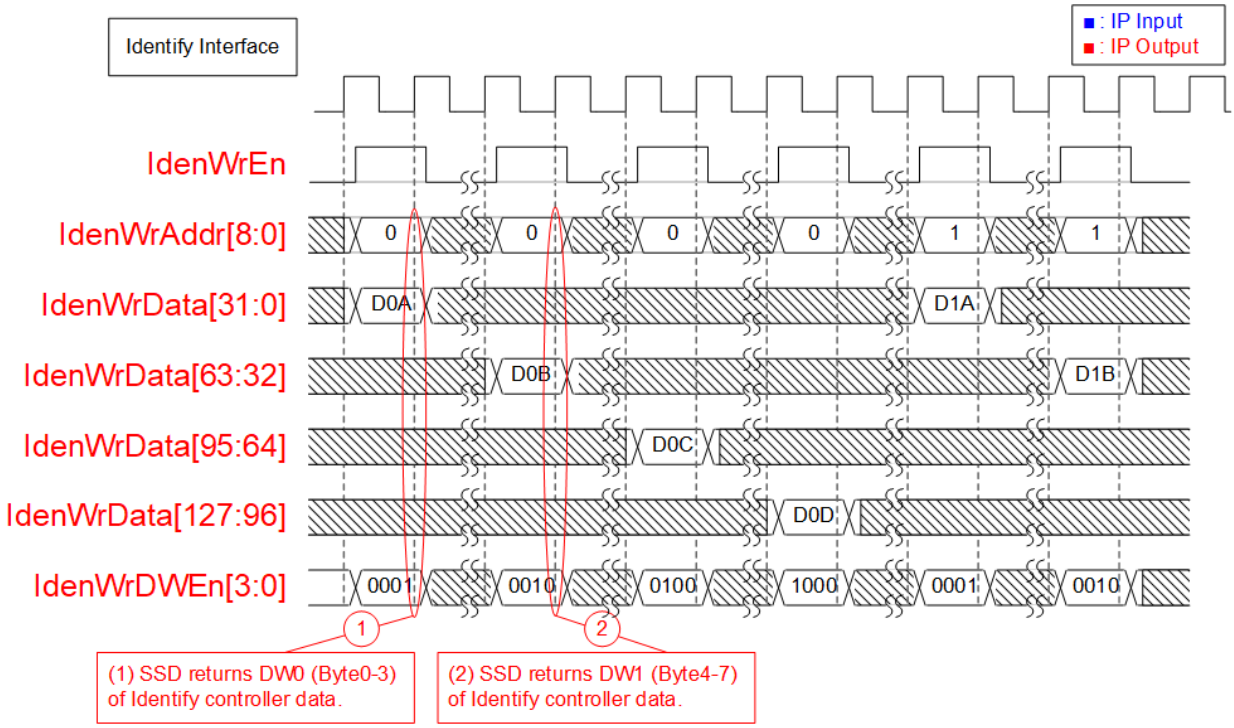

Figure 12: IdenWrDWEn timing diagram

IdenWrDWEn is 4-bit signal to be valid signal of 32-bit data. Some SSDs do not return 4-Kbyte 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 12. IdenWrDWEn[0], [1], [2], and [3] corresponds to IdenWrData[31:0], [63:32], [95:64], and [127:96], 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, raNVMe IP and SSD are inactive until the system is powered down.

Note: If the SSD is powered down without Shutdown command, the total count of unsafe shutdowns (returned data of SMART command) is increased.

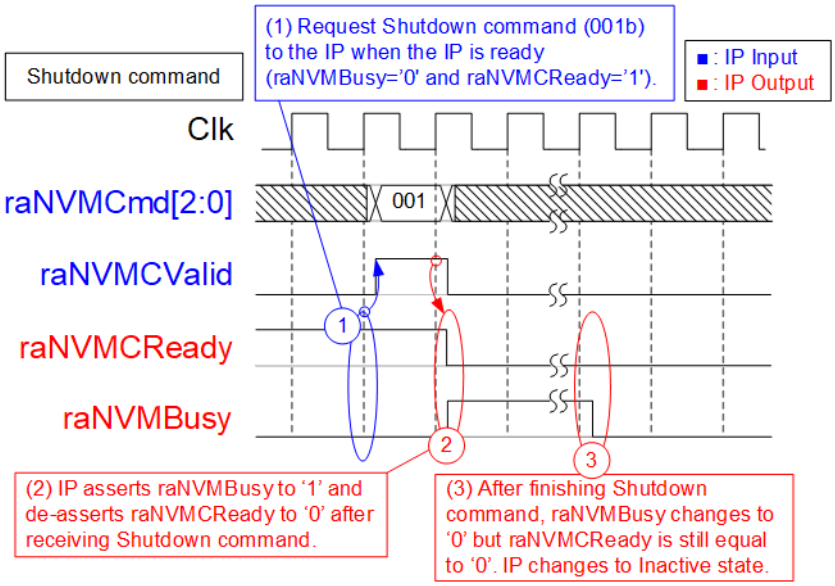

Figure 13: Shutdown command timing diagram

The details when running Shutdown command is shown as follows.

1) Before sending the command request, the IP must be Idle (raNVMBusy='0' and raNVMCReady='1'). To send Shutdown command, user asserts raNVMValid to '1' along with raNVMCmd = 001b.
2) When raNVMe IP receives Shutdown command, the IP asserts raNVMBusy to '1' and de-asserts raNVMCReady to '0'.
3) raNVMBusy is de-asserted to '0' after the SSD is completely shut down. However, raNVMCReady is still equal to '0' and the IP does not receive any command afterwards.

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. After that, the SMART data is returned via CtmRAM port as shown in Figure 14.

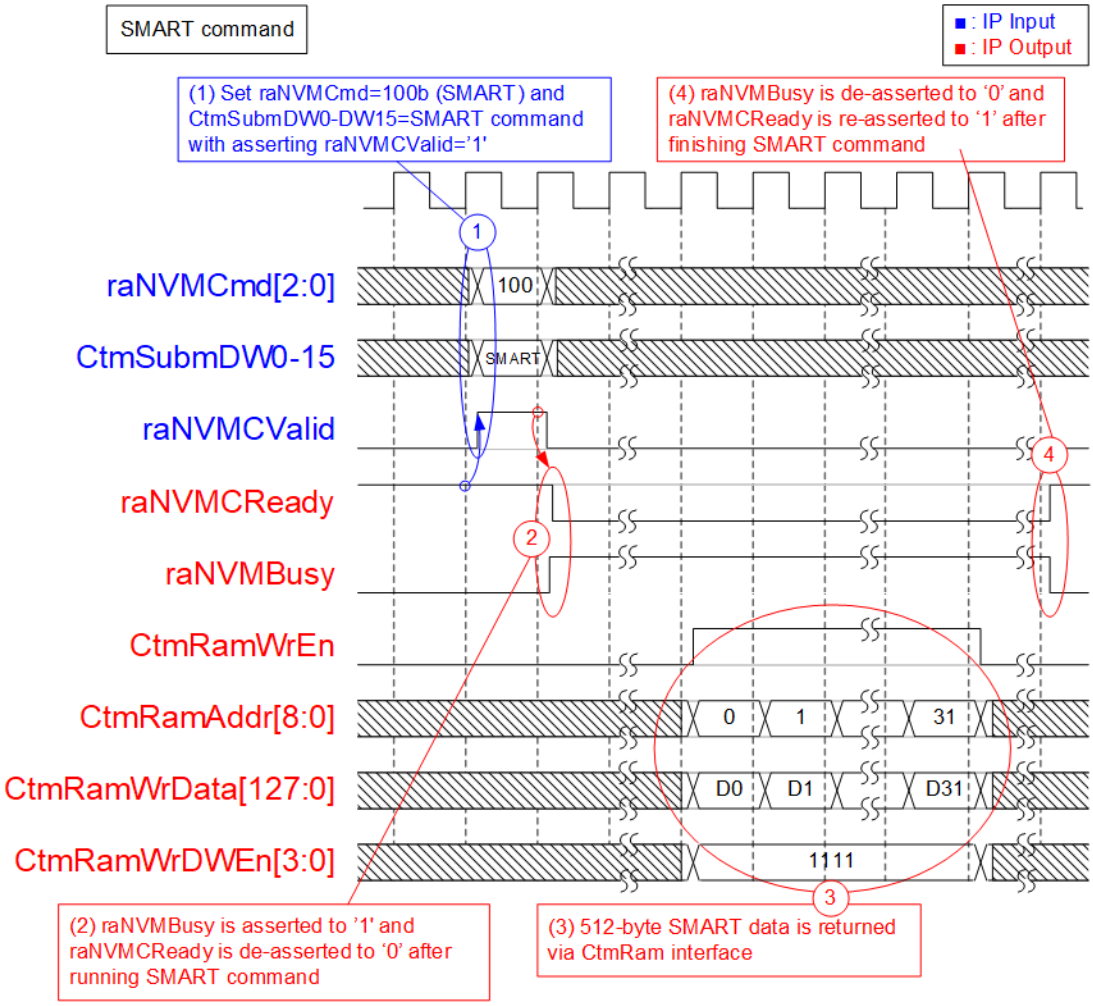

Figure 14: SMART command timing diagram

The details when running SMART command is shown as follows.

1) Before sending the command request, the IP must be ready to receive command (raNVMBusy='0' and raNVMCReady='1'). Next, asserts raNVMCValid with raNVMCmd=100b and valid CtmSubmDW0-DW15 which must be set by following value for SMART command.
   CtmSubmDW0 = 0x0000_0002
   CtmSubmDW1 = 0xFFFF_FFFF
   CtmSubmDW2 – CtmSubmDW5 = 0x0000_0000
   CtmSubmDW6 = 0x2000_0000
   CtmSubmDW7 – CtmSubmDW9 = 0x0000_0000
   CtmSubmDW10 = 0x007F_0002
   CtmSubmDW11 – CtmSubmDW15 = 0x0000_0000
2) raNVMe IP accepts SMART command by asserting raNVMBusy to '1' and de-asserting raNVMCReady to '0'.
3) 512-byte SMART data is returned on CtmRamWrData signal with asserting CtmRamWrEn to '1'. CtmRamAddr is equal to 0-31 to be data index of 512-byte data. When CtmRamAddr=0, byte0-15 of SMART data is returned on CtmRamWrData. CtmRamWrDWEn is Dword enable for each 32-bit CtmRamWrData. If CtmRamWrDWEn=1111b, all 128-bit of CtmRamWrData are valid.
4) After SMART command operation is finished, raNVMBusy is de-asserted to '0' and raNVMCReady is asserted to '1' to receive the next command.

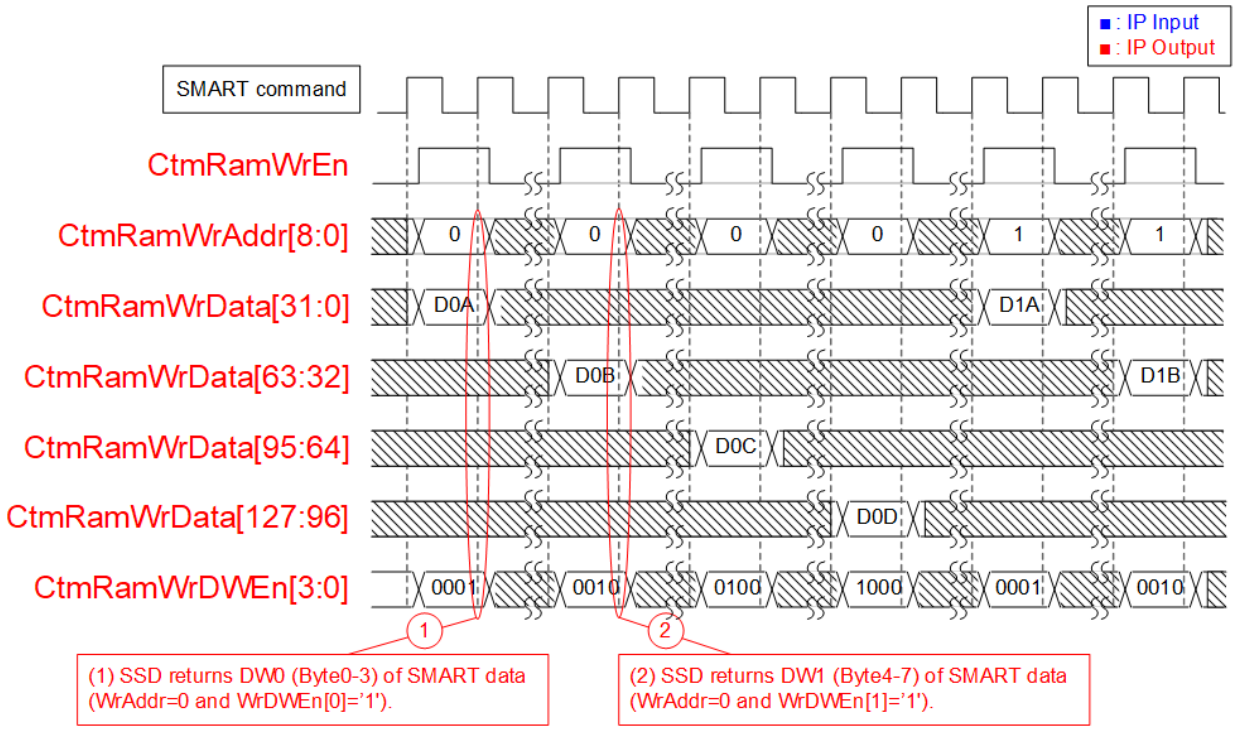

Figure 15: CtmRamWrDWEn timing diagram

Similar to Identify command, some SSDs do not return 512-byte data continuously but returns 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], [2], and [3] corresponds to CtmRamWrData[31:0], [63:32], [95:64], and [127:96], 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.

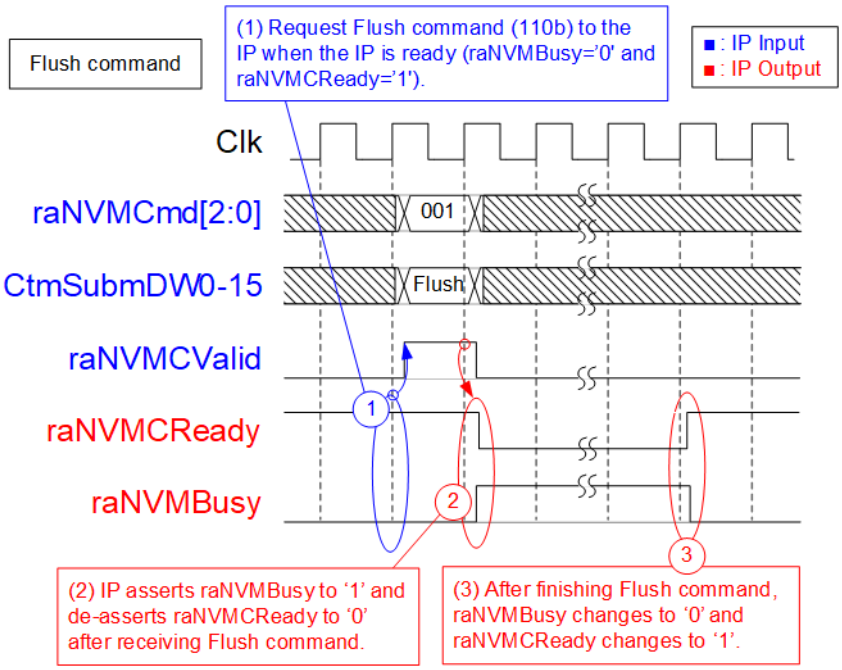

Figure 16: Flush command timing diagram

The details for running Flush command are shown as follows.

1) Before sending the command request, the IP must be ready to receive command (raNVMBusy='0' and raNVMCReady='1'). Then, asserts raNVMCValid with raNVMCmd=100b and valid CtmSubmDW0-DW15 which must be set by following value for Flush command.
   CtmSubmDW0 = 0x0000_0000
   CtmSubmDW1 = 0x0000_0001
   CtmSubmDW2 – CtmSubmDW15 = 0x0000_0000
2) raNVMe IP accepts Flush command by asserting raNVMBusy to '1' and de-asserting raNVMCReady to '0'.
3) After finishing Flush command operation, raNVMBusy is de-asserted to '0' and raNVMCReady is asserted to '1' to receive the next command.

Error

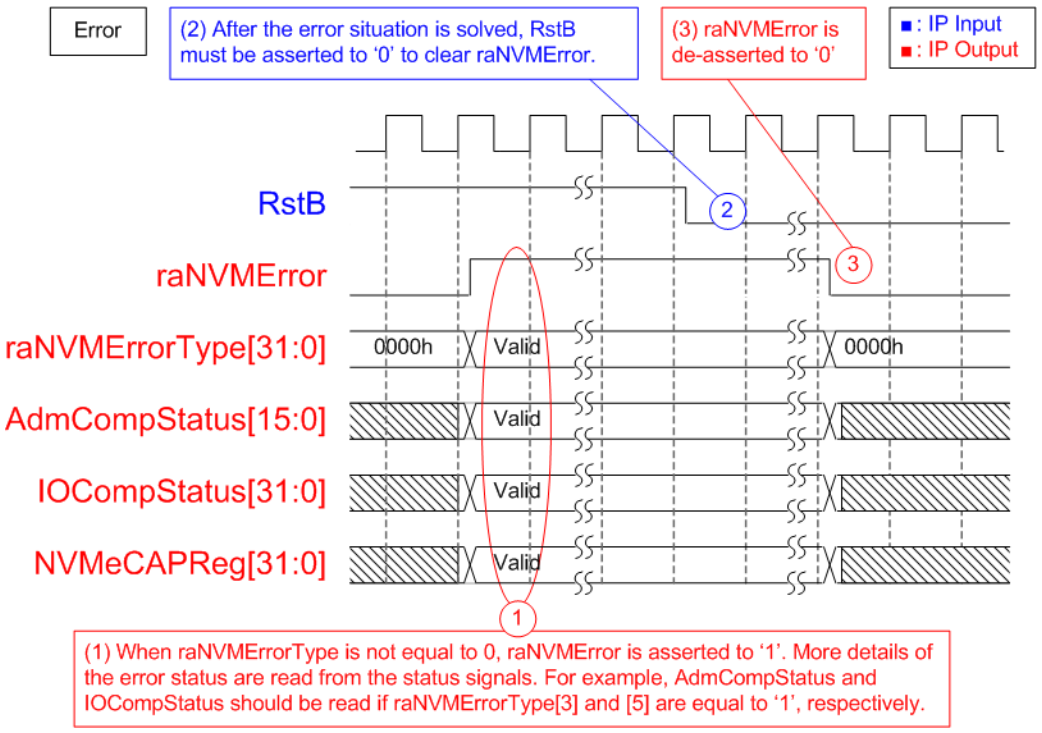

Figure 17: Error flag timing diagram

When the error is found while running initialization process or operating some commands, raNVMeError flag is asserted to '1'. raNVMErrorType is read to check the error type. NVMeCAPReg, AdmCompStatus, and IOCompStatus are also valid for monitoring error details after raNVMError 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 AdmCompStatus and IOCompStatus. When bit[3] of raNVMErrorType is asserted, read AdmCompStatus to check more details. When bit[5] of raNVMErrorType is asserted, read IPCompStatus to check more details.

The raNVMError can be cleared by RstB signal only. After the failure is solved, RstB is asserted to '0' to clear the error flag.

Verification Methods

The raNVMe IP Core functionality was verified by simulation and also proved on real board design by using KCU105/ZCU106/KCU116/VCU118 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 Gatway 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
2.0	8-Nov-2022	Add raNVMrPause signal to pause reading data and support VCU118 board
1.3	15-Jun-2022	Support KCU116 and update resource of URAM
1.2	8-Jan-2021	Support ZCU106
1.1	25-Aug-2020	Correct features and update company info
1.0	11-Aug-2020	Initial Release