

POLITECNICO DI TORINO

Master degree course in Electronic Engineering

Master Degree Thesis
in
Integrated System Architecture

**Computational Storage Drive based
on OpenSSD**



Supervisor

Candidate

Giuseppe DONGIOVANNI
MANCINO

ANNO ACCADEMICO 2019-2020

Contents

List of Tables	3
List of Figures	4
1 Computational Storage Project	7
1.1 Cosmos+ OpenSSD Project	7
1.1.1 Overview	7
1.1.2 Project Adaptation	11
1.1.3 Functionality Tests	13
1.1.4 Performance Tests	13
1.1.5 Optimized Firmware Version	16
1.2 Computational Storage Development	20
1.2.1 32-bit AXI DMA	20
1.2.2 First Prototype	23
1.2.3 128-bit AXI DMA	24
1.2.4 Second Prototype	27
2 Conclusion	41
Bibliography	43

List of Tables

1.1	Set Parameter - Feature Identifiers Assignation	35
2.1	Iometer Sequential Read Test - block size 4kB	42
2.2	Latency Sequential Read Test - block size 4kB	42

List of Figures

1.1	OpenSSD project History - Cosmos+ OpenSSD 2017 Tutorial	8
1.2	Cosmos+ OpenSSD project System Overview - Cosmos+ OpenSSD 2017 Tutorial	8
1.3	Cosmos+ OpenSSD project System Design	9
1.4	Nand Modules Organization - Cosmos+ OpenSSD 2017 Tutorial	10
1.5	Command Priority - Cosmos+ OpenSSD 2017 Tutorial	11
1.6	Firmware Overall Sequence - Cosmos+ OpenSSD 2017 Tutorial	11
1.7	First Adaptation of the Cosmos+ OpenSSD project System Design	12
1.8	Functionality test - Power-up, Partitioning and Reset - Development PC	14
1.9	Functionality test - Power-up - Host PC	15
1.10	Functionality test - Formatting operation - Host PC	15
1.11	Functionality test - Data Correctness - Host PC	15
1.12	Functionality test - NVMe Identify Command - Host PC	16
1.13	Functionality test - NVMe Namespace List - Host PC	16
1.14	Performance test - dd function, write and read - Base Version	16
1.15	Performance test - Iometer Sequential Read transfer size = 4k, block size = 4k - Base Version	17
1.16	Performance test - Iometer Sequential Read transfer size = 16k, block size = 4k - Base Version	17
1.17	Performance test - Read Latency - Base Version	18
1.18	Terminal Error - head.autoDmaRx stuck	18
1.19	Waveform Error - head.autoDmaRx stuck	18
1.20	Waveform Error - Submitted DMA Request	18
1.21	Performance test - dd function, write and read - Optimized Version	19
1.22	Performance test - Iometer Sequential Read transfer size = 4k, block size = 4k - Optimized Version	19
1.23	Performance test - Iometer Sequential Read transfer size = 16k, block size = 4k - Optimized Version	19
1.24	General Architecture of a FPGA-based Computational Storage	20
1.25	32bit AXI DMA Adaptation of the Cosmos+ OpenSSD project System Design	21

1.26	Performance test - dd function, write and read - 32bit AXI DMA Version	21
1.27	Performance test - Iometer Sequential Read transfer size = 4k, block size = 4k - 32bit AXI DMA Adaptation	21
1.28	Performance test - Iometer Sequential Read transfer size = 16k, block size = 4k - 32bit AXI DMA Adaptation	22
1.29	Performance test - Read Latency - 32bit AXI DMA Adaptation	22
1.30	System Design of the first version of the Computational Storage based on the Cosmos+ OpenSSD project	23
1.31	FSM of the first version of the Hardware Accelerator	24
1.32	NVMe Set Parameter Admin Command - Addition of 0x7 to a 32-bit sequence - Host and Developer PCs	24
1.33	Performance test - dd function, write and read - Computational storage first prototype	25
1.34	Performance test - Iometer Sequential Read transfer size = 4k, block size = 4k - Computational storage first prototype	25
1.35	Performance test - Iometer Sequential Read transfer size = 16k, block size = 4k - Computational storage first prototype	25
1.36	Performance test - Read Latency - Computational storage first prototype	26
1.37	Performance test - dd function, write and read - 128bit AXI DMA Version	26
1.38	Performance test - Iometer Sequential Read transfer size = 4k, block size = 4k - 128bit AXI DMA Adaptation	27
1.39	Performance test - Iometer Sequential Read transfer size = 16k, block size = 4k - 128bit AXI DMA Adaptation	27
1.40	Performance test - Read Latency - 128bit AXI DMA Adaptation	28
1.41	General diagram of the Hardware Accelerator	28
1.42	FSMs of the second version of the Hardware Accelerator	29
1.43	CTR Mode - NIST, Recommendation for Block Cipher Modes of Operation: Methods and Techniquesm	31
1.44	Verilog Test-bench - AES-128 Encryption	32
1.45	Verilog Test-bench - AES-128 Decryption	32
1.46	Verilog Test-bench - AES-256 Encryption	33
1.47	Verilog Test-bench - AES-256 Decryption	33
1.48	Module execution - AES-128 Terminal	33
1.49	Module execution - AES-128 Configuration	34
1.50	Module execution - AES-128 Encryption	34
1.51	Module execution - AES-256 Terminal	34
1.52	Module execution - AES-256 Configuration	34
1.53	Module execution - AES256 Encryption	35
1.54	General Execution - AES-128 Host PC Terminal	36

1.55	General Execution - AES-128 Developer PC Terminal	36
1.56	General Execution - AES-128 Encryption Waveform	36
1.57	General Execution - AES-256 Host PC Terminal	37
1.58	General Execution - AES-256 Developer PC Terminal	37
1.59	General Execution - AES-256 Encryption Waveform	37
1.60	Performance test - dd function, write and read - 128bit AXI DMA Version	38
1.61	Performance test - Iometer Sequential Read transfer size = 4k, block size = 4k - Computational Storage	38
1.62	Performance test - Iometer Sequential Read transfer size = 16k, block size = 4k - Computational Storage	39
1.63	Performance test - Read Latency - Computational Storage	39

Chapter 1

Computational Storage Project

In this chapter all the steps necessary to build an FPGA-based NVMe Computational Storage are going to be analyzed. However, due its complexity, it is necessary to first obtain the NVMe communication interface: an NVMe Controller. Therefore, the open source project Cosmos+ OpenSSD has been chosen as the basis for our goal due to matching the requirements.

1.1 Cosmos+ OpenSSD Project

The first step consists of analysis and adaptation of the OpenSSD project from the original custom board to the Xilinx Zynq-7000 SoC ZC706.

1.1.1 Overview

Cosmos OpenSSD is an open source and FPGA-based SSD controller project that has been developed since 2014 by the HYU ENC Lab of the Hanyang University in South Korea, with research and education purposes[1].

A first version of the project was based on Indilinx Barefoot, a SoC over SATA2.

The project version that will be analyzed is the Cosmos+ OpenSSD: developed in 2016, this version of the SSD controller supports the NVMe protocol. The project has been developed using Xilinx Developer Tools, Vivado Design Suite and SDK.

The custom Cosmos+ FPGA board has the following main features:

- FPGA Xilinx Zynq-7000 with a Dual ARM Cortex-A9 1GHz Core;
 - 1GB of DDR3;
 - AXI4-lite bus width of 32 bits;
 - AXI4 bus width of 64 bits;
- dual PCIe Gen2 x8 End-Points (Cabled PCIe Interface);

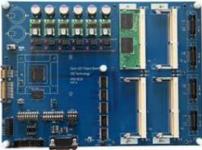
	Jasmine OpenSSD	Cosmos OpenSSD	Cosmos+ OpenSSD
Released in	2011	2014	2016
Main Board			
SSD Controller	Indilinx Barefoot (SoC)	HYU Tiger3 (FPGA)	HYU Tiger4 (FPGA)
Host Interface	SATA2	PCIe Gen2 4-lane (AHCI)	PCIe Gen2 8-lane (NVMe)
Maximum Capacity	128 GB (32 GB/module)	256 GB (128 GB/module)	2 TB (1 TB/module)
NAND Data Interface	SDR (Asynchronous)	NVDDR (Synchronous)	NVDDR2 (Toggle)
ECC Type and Strength	BCH, 16 bits/512 B	BCH, 32 bits/2 KB	BCH, 26 bits/512 B

Figure 1.1: OpenSSD project History - Cosmos+ OpenSSD 2017 Tutorial

- additional interfaces (JTAG,USB,Ethernet);
- up to 2 NAND Flash Modules, with 8 flash packages slot each.

In Figure 1.2 it is illustrated the internal system overview of the project.

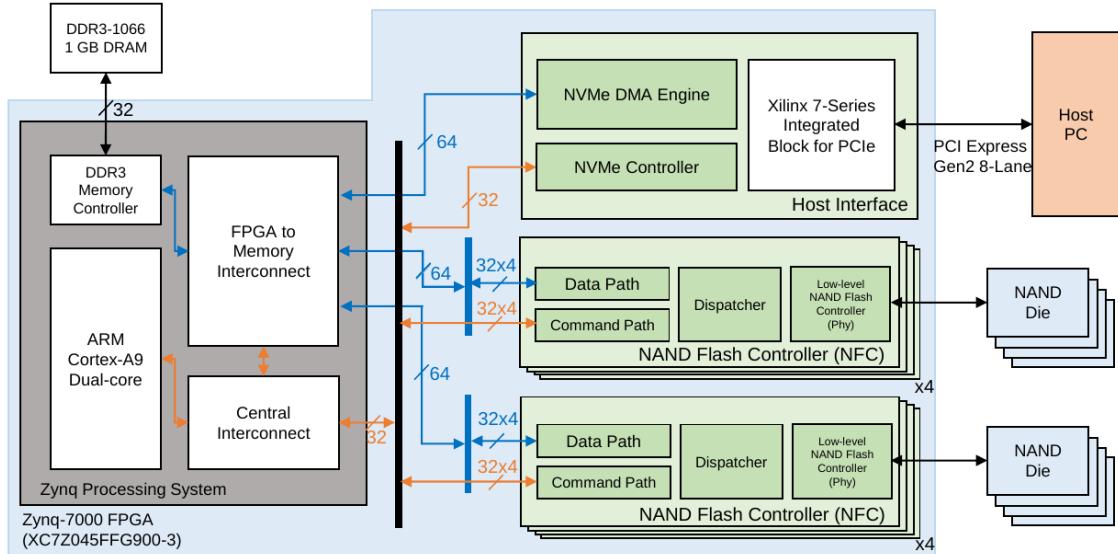


Figure 1.2: Cosmos+ OpenSSD project System Overview - Cosmos+ OpenSSD 2017 Tutorial

The Zynq processor is connected to the host through the Host Interface, called NVMe Host Controller, which is responsible of:

- handling of the data from the host to the buffer with a DMA engine;

- automated completion of the NVMe IO Command, without involving the Flash Transition Layer (FTL), that will be described later.

The NAND Flash Controller is the interface between the NAND Flash and the processor. It consists, as shown in the system block design in Figure 1.3, of three different hardware IP blocks:

- Tiger4 NSC;
- Tiger4 Shared KES;
- V2NFC.

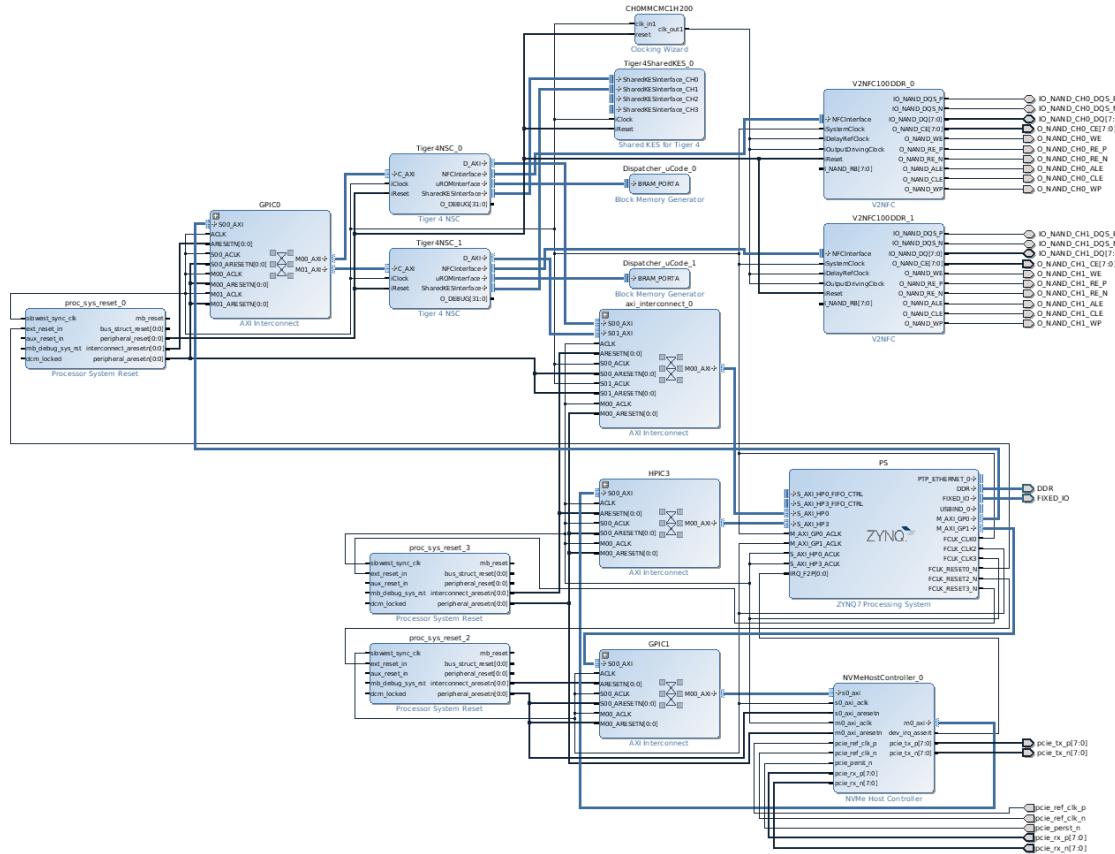


Figure 1.3: Cosmos+ OpenSSD project System Design

The IP Tiger4 NSC is the responsible of the handling of command and data from the processing system: the commands, consisting of information such as source and destination of the operation, are written by the firmware driver in the Tiger4 NSC registers and then elaborated.

The data, instead, undergo different manipulations: in particular they pass through a module responsible of the Error Detection and Correction, the Tiger4 Shared KES.

Finally, the data are handled by the V2NFC block, that physically performs the low-level I/O operations in the NAND Flash Modules.

The operations have to be scheduled in order to be performed on the right SSD package and die. As shown in Figure 1.4, each NAND module has up to 4 available channels, one every two packages, and each channel has 8 maximum ways, corresponding to the maximum number of connected dies. The number of channels is equal to the number of NAND Flash Controllers.

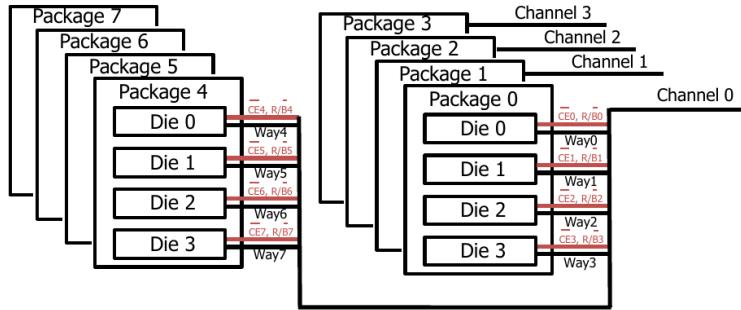


Figure 1.4: Nand Modules Organization - Cosmos+ OpenSSD 2017 Tutorial

In the first version of the project, Cosmos, the way scheduling was managed by the NFC block, while the channel one by the firmware Flash Transition Layer (FTL): with the Cosmos+ one, both channel and way scheduling is managed by the FTL, providing more flexibility.

The other main features of the FTL are:

- Least Recently Used (LRU) data buffer management;
- priority command scheduling, as shown in Figure 1.5, with the aim of enhancing the multi-channel and way parallelism;
- on demand garbage collection, triggered only when there is no more free user block in each die;

The garbage collection is needed to recover free blocks for write requests: a victim block with invalid data is selected, then the valid data are copied in a free block while the victim one is erased.

However, while supporting the garbage collection, the firmware does not support the wear leveling of the flash memory.

A schematic description of the firmware execution is shown in Figure 1.6.

In order to be executed, a received IO command is first transformed in Slice Requests, which number depends on the number of logic blocks requested. Then

Command	Priority
LLSCommand_RxDMA	0
LLSCommand_TxDMA	0
V2FCommand_StatusCheck	1
V2FCommand_ReadPageTrigger	2
V2FCommand_BlockErase	3
V2FCommand_ProgramPage	4
V2FCommand_ReadPageTransfer	5

Figure 1.5: Command Priority - Cosmos+ OpenSSD 2017 Tutorial

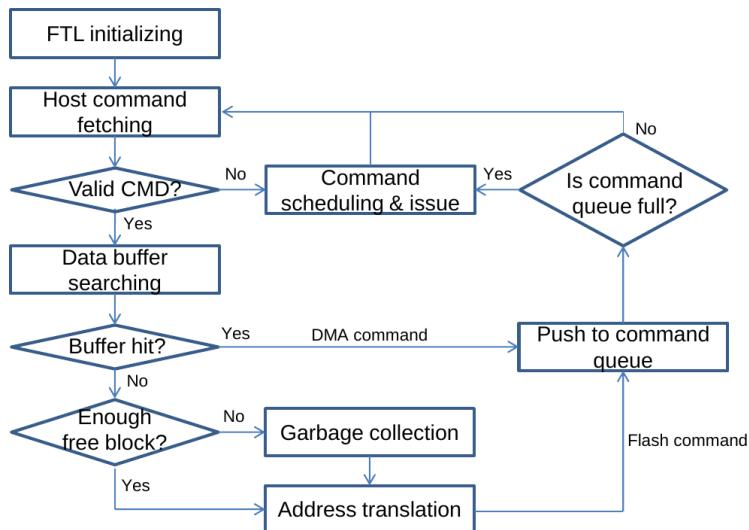


Figure 1.6: Firmware Overall Sequence - Cosmos+ OpenSSD 2017 Tutorial

each Slice Request undergoes a second transformation in DMA and, if there is no buffer hit for read operations, NAND Requests. These requests are organized in different queues: one for the free requests, three for the requests that are going to be executed, one for each aforementioned type of requests, and two for the blocked requests, either for buffer or row address dependencies. Then the requests are, if not blocked, finally scheduled and executed.

1.1.2 Project Adaptation

The available platform is the Xilinx ZC706: in comparison with the original custom board, it has:

- same FPGA Zynq-7000 SoC;

- 4-line Gen2 PCIe Connector, instead of the 8-lane of the custom board;
- no NAND module.

The first step to adapt the project is to modify the hardware¹: there is no NAND module, therefore the entire NAND Flash Controller hardware is not necessary.

At the same time, the NVMe Host Controller has to be changed from the 8-lane PCIe to the 4-lane one: this can be done by modifying the configuration of the Xilinx PCIe Core IP. The result is shown in Figure 1.7. Likewise, the constraint files have to be modified or removed, in order to match the pinout of the Xilinx ZC706[2].

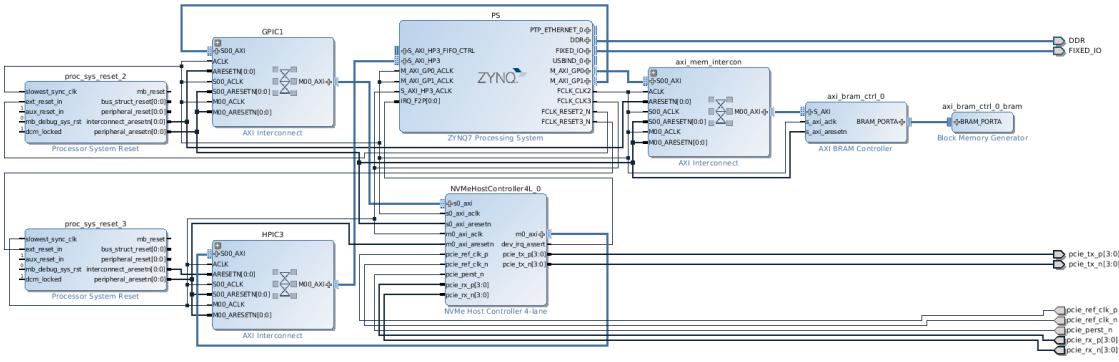


Figure 1.7: First Adaptation of the Cosmos+ OpenSSD project System Design

A BRAM Controller has been added to provide a destination address for the firmware channel, substituting the Tiger4NSC one: however, it has no active role in the operations.

After exporting the new hardware file (.hdf), it is necessary to create a new project with the new platform specifications.

Some modifications have also been made to the firmware:

- allocation of the memory arrays <MemSpace> in the DDR: the storage capacity is of 64 MB, due to the DDR already being used by the firmware FTL;
- different organization of the memory management unit (MMU) table and the memory mapping, given the presence of the memory array;
- variation to the memory dimensions, number of channel and way;
- bypass of the status check and ECC functions;

¹This modified project and all the following ones have been uploaded on the GitHub repository <https://github.com/giuseppedongiovanni/nvme_comp_storage>

- replacement of the NAND operations, represented by the write operation of the commands in the Tiger4 registers, with the MemCopy function.

The memory array `<MemSpace>` is the replacement for the SSD: however, the available storage is much smaller than what the firmware expects. Varying the FTL configuration parameters, that should be left untouched, is necessary to avoid that important memory location useful for the firmware execution are corrupted.

1.1.3 Functionality Tests

The first test that has been carried out is the functionality one, consisting of a routine of power-up, partitioning and reset from both developer and host side, as shown respectively in Figures 1.8, 1.9 and 1.10. Then the data correctness, that corresponds to writing a file and reading it back, is verified in Figure 1.11 by the matching MD5 values (practically the digital fingerprint of a file).

Finally, it is possible to send some NVMe Admin Commands to verify the presence of the namespace and, as well, obtain information about the device. As it can be seen in Figure 1.13, the namespace id is set to 14740, equal to the storage capacity in terms of NVMe blocks (4096 bytes): this namespace, defined by the project creators, has been attached by the controller to the NSID 1, being the device seen as `<nvme0n1>`.

1.1.4 Performance Tests

Different performance tests are carried out: `<dd>` function and the software Iometer are used to evaluate IOPS and bandwidth; to measure the latency instead, timers are used both in the firmware and in a c-file on the host side.

The Figures 1.14 to 1.16 refer to the performed tests for the read operation.

The results obtained are unexpected: performing operations from a DDR to an host device through a PCIe bus should imply very high performance, in the order of GB/s for the bandwidth. Instead, the obtained one is around 100 MB/s, with only 7000 IOPS.

More information can be obtained from the analysis of the latency test results, shown in Figure 1.17.

The pie chart is divided in two main parts: the green one is related to the time spent in the device firware, while the orange one includes all the other contributions, in particular the host, the PCIe bus and the NVMeHostController of the device.

The firmware execution takes up the 73% of the total latency: in particular, the `<MemCopy>` function employs more than half of this time period to be performed, slowing the entire execution. The obtained speed is of about 150 MB/s, while a single-port DDR3, with a 32-bit bus-width at 533 MHz, has a maximum theoretical bandwidth of 4 GB/s.

```

Hello COSMOS+ OpenSSD !!!
!!! Wait until FTL reset complete !!!
[ NAND device reset complete. ]
Press 'X' to re-make the bad block table.
[ bad block table of ch 0 way 0 exists. ]
[ bad blocks of ch 0 way 0 are checked. ]
new bad block table
Bad block remapping start...
No reserved block - Ch 0 Way 0 virtualBlock 0 is bad block
Bad block remapping end
Erase User block space...wait for a minute...
Done.
[ storage capacity 57 MB ]
[ ftl configuration complete. ]

FTL reset complete!!!
Turn on the host PC
PCIe Link: 1
PCIe Link: 0

NVMe reset!!!
PCIe Link: 1
PCIe Bus Master: 1
PCIe Bus Master: 0
PCIe IRQ Disable: 0
PCIe IRQ Disable: 1
PCIe Bus Master: 1
PCIe IRQ Disable: 0
PCIe MSI Enable: 1, 0x0
NVME CC.EN: 1

NVMe ready!!!
Done Admin Command OPC: 6
num of queue 70007
Set Feature FID:7
PCIe MSI Enable: 0, 0x0
PCIe IRQ Disable: 0
PCIe MSI Enable: 1, 0x0
one Admin Command OPC: 9
create cq: 0x00000003, 0x00FF0001
Done Admin Command OPC: 5
Done Admin Command OPC: 5
create sq: 0x00010001, 0x00FF0001
Done Admin Command OPC: 1
Done Admin Command OPC: 6
Done Admin Command OPC: C
Done Admin Command OPC: 6
delete sq: 0x00000001
Done Admin Command OPC: 0
delete cq: 0x00000001
NVME CC.SHN: 1
one Admin Command OPC: 4

NVMe shutdown!!!
PCIe MSI Enable: 0, 0x0
PCIe IRQ Disable: 1
PCIe Bus Master: 0
PCIe Link: 0
NVME CC.EN: 0
NVME CC.SHN: 0

NVMe disable!!!

```

Figure 1.8: Functionality test - Power-up, Partitioning and Reset - Development PC

The cause of this problem has to be found in the project itself: both the processing system and the NVMeHostController IP are running with a heavy memory access loading through the same port of the DDR, that for this reason has become the bottleneck of the project.

On the other hand, no write test are available: if the device is stressed with long or continuous write operations, the DMA of the NVMe Host Controller freezes, resulting in a Timeout Abort Error on the host.

In Figure 1.18 different variables were printed in the terminal to backtrack the cause of the error: in particular, head.autoDmaRx is the hardware counter of the completed DMA request, while tail.autoDmaRx is the software counter of the submitted DMA request: when the two counters coincide, the DMA operation is completed. It is possible to see that the DMA is stuck at head.autoDmaRx = 0xAA, although other 2 requests are present in the queue, being tail.autoDmaRx = 0xAC.

1.1 – Cosmos+ OpenSSD Project

```

File Edit View Search Terminal Help
reds@reds:~$ ls /dev
loop0 loop1 loop2 loop3 loop4 loop5 loop6 loop7
nvmed0 rtkill tty0 tty23 tty35 tty48 tty62
loop8 nvmed1 rtc tty10 tty22 tty36 tty58 tty53b vcs2 vcs3
loop9 nvmed2 loop10 loop11 loop12 loop13 loop14 loop15 loop16
nvmed3 sda tty11 tty21 tty37 tty59 tty71 vcs3 vcs4
loop17 nvmed4 loop18 loop19 loop20 loop21 loop22 loop23 loop24
nvmed5 sdb tty12 tty22 tty38 tty59 tty53c vcs5 vcs6
loop25 nvmed6 loop26 loop27 loop28 loop29 loop30 loop31 loop32
nvmed7 sdc tty13 tty23 tty39 tty60 tty72 vcs7 vcs8
loop33 nvmed8 loop34 loop35 loop36 loop37 loop38 loop39 loop40
nvmed9 sdd tty14 tty24 tty40 tty61 tty73 vcs9 vcs10
loop41 nvmed10 loop42 loop43 loop44 loop45 loop46 loop47 loop48
nvmed11 sde tty15 tty25 tty41 tty62 tty74 vcs11 vcs12
loop49 nvmed12 loop50 loop51 loop52 loop53 loop54 loop55 loop56
nvmed13 sdd2 tty16 tty26 tty42 tty63 tty75 vcs13 vcs14
loop57 nvmed14 loop58 loop59 loop60 loop61 loop62 loop63 loop64
nvmed15 sdd3 tty17 tty27 tty43 tty64 tty76 vcs15 vcs16
loop65 nvmed16 loop66 loop67 loop68 loop69 loop70 loop71 loop72
nvmed17 sdd4 tty18 tty28 tty44 tty65 tty77 vcs17 vcs18
loop73 nvmed18 loop74 loop75 loop76 loop77 loop78 loop79 loop80
nvmed19 sdd5 tty19 tty29 tty45 tty66 tty78 vcs19 vcs20
loop81 nvmed20 loop82 loop83 loop84 loop85 loop86 loop87 loop88
nvmed21 sdd6 tty20 tty30 tty46 tty67 tty79 vcs21 vcs22
loop89 nvmed22 loop90 loop91 loop92 loop93 loop94 loop95 loop96
nvmed23 sdd7 tty21 tty31 tty47 tty68 tty80 vcs23 vcs24
loop97 nvmed24 loop98 loop99 loop100 loop101 loop102 loop103 loop104
nvmed25 sdd8 tty22 tty32 tty48 tty69 tty81 vcs25 vcs26
loop105 nvmed26 loop106 loop107 loop108 loop109 loop110 loop111 loop112
nvmed27 sdd9 tty23 tty33 tty49 tty70 vcs27 vcs28
loop113 nvmed28 loop114 loop115 loop116 loop117 loop118 loop119 loop120
nvmed29 sdd10 tty24 tty34 tty50 vcs29 vcs30
loop121 nvmed30 loop122 loop123 loop124 loop125 loop126 loop127 loop128
nvmed31 sdd11 tty25 tty35 tty51 vcs31 vcs32
loop135 nvmed32 loop136 loop137 loop138 loop139 loop140 loop141 loop142
nvmed33 sdd12 tty26 tty36 tty52 vcs33 vcs34
loop143 nvmed34 loop144 loop145 loop146 loop147 loop148 loop149 loop150
nvmed35 sdd13 tty27 tty37 tty53 vcs35 vcs36
loop151 nvmed36 loop152 loop153 loop154 loop155 loop156 loop157 loop158
nvmed37 sdd14 tty28 tty38 tty54 vcs37 vcs38
loop159 nvmed38 loop160 loop161 loop162 loop163 loop164 loop165 loop166
nvmed39 sdd15 tty29 tty39 tty55 vcs39 vcs40
loop167 nvmed40 loop168 loop169 loop170 loop171 loop172 loop173 loop174
nvmed41 sdd16 tty30 tty40 tty56 vcs41 vcs42
loop175 nvmed42 loop176 loop177 loop178 loop179 loop180 loop181 loop182
nvmed43 sdd17 tty31 tty41 tty57 vcs43 vcs44
loop183 nvmed44 loop184 loop185 loop186 loop187 loop188 loop189 loop190
nvmed45 sdd18 tty32 tty42 tty58 vcs45 vcs46
loop191 nvmed46 loop192 loop193 loop194 loop195 loop196 loop197 loop198
nvmed47 sdd19 tty33 tty43 tty59 vcs47 vcs48
loop199 nvmed48 loop200 loop201 loop202 loop203 loop204 loop205 loop206
nvmed49 sdd20 tty34 tty44 tty60 vcs49 vcs50
loop207 nvmed50 loop208 loop209 loop210 loop211 loop212 loop213 loop214
nvmed51 sdd21 tty35 tty45 tty61 vcs51 vcs52
loop215 nvmed52 loop216 loop217 loop218 loop219 loop220 loop221 loop222
nvmed53 sdd22 tty36 tty46 tty62 vcs53 vcs54
loop223 nvmed54 loop224 loop225 loop226 loop227 loop228 loop229 loop220
nvmed55 sdd23 tty37 tty47 tty63 vcs55 vcs56
loop228 nvmed56 loop229 loop230 loop231 loop232 loop233 loop234 loop235
nvmed57 sdd24 tty38 tty48 tty64 vcs57 vcs58
loop236 nvmed58 loop237 loop238 loop239 loop240 loop241 loop242 loop243
nvmed59 sdd25 tty39 tty49 tty65 vcs59 vcs60
loop244 nvmed60 loop245 loop246 loop247 loop248 loop249 loop250 loop251
nvmed61 sdd26 tty40 tty50 tty66 vcs61 vcs62
loop252 nvmed62 loop253 loop254 loop255 loop256 loop257 loop258 loop259
nvmed63 sdd27 tty41 tty51 tty67 vcs63 vcs64
loop256 nvmed64 loop257 loop258 loop259 loop260 loop261 loop262 loop263
nvmed65 sdd28 tty42 tty52 tty68 vcs65 vcs66
loop264 nvmed66 loop265 loop266 loop267 loop268 loop269 loop270 loop271
nvmed67 sdd29 tty43 tty53 tty69 vcs67 vcs68
loop272 nvmed68 loop273 loop274 loop275 loop276 loop277 loop278 loop279
nvmed69 sdd30 tty44 tty54 tty70 vcs69 vcs70
loop277 nvmed70 loop278 loop279 loop280 loop281 loop282 loop283 loop284
nvmed71 sdd31 tty45 tty55 tty71 vcs71 vcs72
loop285 nvmed72 loop286 loop287 loop288 loop289 loop290 loop291 loop292
nvmed73 sdd32 tty46 tty56 tty72 vcs73 vcs74
loop293 nvmed74 loop294 loop295 loop296 loop297 loop298 loop299 loop290
nvmed75 sdd33 tty47 tty57 tty73 vcs75 vcs76
loop297 nvmed76 loop298 loop299 loop300 loop301 loop302 loop303 loop304
nvmed77 sdd34 tty48 tty58 tty74 vcs77 vcs78
loop305 nvmed78 loop306 loop307 loop308 loop309 loop3010 loop3011 loop3012
nvmed79 sdd35 tty49 tty59 tty75 vcs79 vcs80
loop313 nvmed80 loop314 loop315 loop316 loop317 loop318 loop319 loop310
nvmed81 sdd36 tty50 tty60 tty76 vcs81 vcs82
loop317 nvmed82 loop318 loop319 loop320 loop321 loop322 loop323 loop324
nvmed83 sdd37 tty51 tty61 tty77 vcs83 vcs84
loop325 nvmed84 loop326 loop327 loop328 loop329 loop3210 loop3211 loop3212
nvmed85 sdd38 tty52 tty62 tty78 vcs85 vcs86
loop333 nvmed86 loop334 loop335 loop336 loop337 loop338 loop339 loop3310
nvmed87 sdd39 tty53 tty63 tty79 vcs87 vcs88
loop341 nvmed88 loop342 loop343 loop344 loop345 loop346 loop347 loop3410
nvmed89 sdd40 tty54 tty64 tty80 vcs89 vcs90
loop349 nvmed90 loop350 loop351 loop352 loop353 loop354 loop355 loop3510
nvmed91 sdd41 tty55 tty65 tty81 vcs91 vcs92
loop357 nvmed92 loop358 loop359 loop3510 loop3511 loop3512 loop3513 loop3514
nvmed93 sdd42 tty56 tty66 tty82 vcs93 vcs94
loop365 nvmed94 loop366 loop367 loop368 loop369 loop3610 loop3611 loop3612
nvmed95 sdd43 tty57 tty67 tty83 vcs95 vcs96
loop373 nvmed96 loop374 loop375 loop376 loop377 loop378 loop379 loop3710
nvmed97 sdd44 tty58 tty68 tty84 vcs97 vcs98
loop381 nvmed98 loop382 loop383 loop384 loop385 loop386 loop387 loop3810
nvmed99 sdd45 tty59 tty69 tty85 vcs99 vcs100
loop389 nvmed100 loop390 loop391 loop392 loop393 loop394 loop395 loop3910
nvmed101 sdd46 tty60 tty70 tty86 vcs101 vcs102
loop397 nvmed102 loop398 loop399 loop3910 loop3911 loop3912 loop3913 loop3914
nvmed103 sdd47 tty61 tty71 tty87 vcs103 vcs104
loop405 nvmed104 loop406 loop407 loop408 loop409 loop4010 loop4011 loop4012
nvmed105 sdd48 tty62 tty72 tty88 vcs105 vcs106
loop413 nvmed106 loop414 loop415 loop416 loop417 loop418 loop419 loop4110
nvmed107 sdd49 tty63 tty73 tty89 vcs107 vcs108
loop421 nvmed108 loop422 loop423 loop424 loop425 loop426 loop427 loop4210
nvmed109 sdd50 tty64 tty74 tty90 vcs109 vcs110
loop429 nvmed110 loop430 loop431 loop432 loop433 loop434 loop435 loop4310
nvmed111 sdd51 tty65 tty75 tty91 vcs111 vcs112
loop437 nvmed112 loop438 loop439 loop4310 loop4311 loop4312 loop4313 loop4314
nvmed113 sdd52 tty66 tty76 tty92 vcs113 vcs114
loop445 nvmed114 loop446 loop447 loop448 loop449 loop4410 loop4411 loop4412
nvmed115 sdd53 tty67 tty77 tty93 vcs115 vcs116
loop453 nvmed116 loop454 loop455 loop456 loop457 loop458 loop459 loop4510
nvmed117 sdd54 tty68 tty78 tty94 vcs117 vcs118
loop461 nvmed118 loop462 loop463 loop464 loop465 loop466 loop467 loop4610
nvmed119 sdd55 tty69 tty79 tty95 vcs119 vcs120
loop469 nvmed120 loop470 loop471 loop472 loop473 loop474 loop475 loop4710
nvmed121 sdd56 tty70 tty80 tty96 vcs121 vcs122
loop477 nvmed122 loop478 loop479 loop4710 loop4711 loop4712 loop4713 loop4714
nvmed123 sdd57 tty71 tty81 tty97 vcs123 vcs124
loop485 nvmed124 loop486 loop487 loop488 loop489 loop4810 loop4811 loop4812
nvmed125 sdd58 tty72 tty82 tty98 vcs125 vcs126
loop493 nvmed126 loop494 loop495 loop496 loop497 loop498 loop499 loop4910
nvmed127 sdd59 tty73 tty83 tty99 vcs127 vcs128
loop499 nvmed128 loop4910 loop4911 loop4912 loop4913 loop4914 loop4915 loop4916
nvmed129 sdd60 tty74 tty84 tty100 vcs129 vcs130
loop507 nvmed130 loop508 loop509 loop5010 loop5011 loop5012 loop5013 loop5014
nvmed131 sdd61 tty75 tty85 tty101 vcs131 vcs132
loop515 nvmed132 loop516 loop517 loop518 loop519 loop5110 loop5111 loop5112
nvmed133 sdd62 tty76 tty86 tty102 vcs133 vcs134
loop523 nvmed134 loop524 loop525 loop526 loop527 loop528 loop529 loop5210
nvmed135 sdd63 tty77 tty87 tty103 vcs135 vcs136
loop531 nvmed136 loop532 loop533 loop534 loop535 loop536 loop537 loop5310
nvmed137 sdd64 tty78 tty88 tty104 vcs137 vcs138
loop539 nvmed138 loop5310 loop5311 loop5312 loop5313 loop5314 loop5315 loop5316
nvmed139 sdd65 tty79 tty89 tty105 vcs139 vcs140
loop547 nvmed140 loop548 loop549 loop5410 loop5411 loop5412 loop5413 loop5414
nvmed141 sdd66 tty80 tty90 tty106 vcs141 vcs142
loop555 nvmed142 loop556 loop557 loop558 loop559 loop5510 loop5511 loop5512
nvmed143 sdd67 tty81 tty91 tty107 vcs143 vcs144
loop563 nvmed144 loop564 loop565 loop566 loop567 loop568 loop569 loop5610
nvmed145 sdd68 tty82 tty92 tty108 vcs145 vcs146
loop571 nvmed146 loop572 loop573 loop574 loop575 loop576 loop577 loop5710
nvmed147 sdd69 tty83 tty93 tty109 vcs147 vcs148
loop579 nvmed148 loop5710 loop5711 loop5712 loop5713 loop5714 loop5715 loop5716
nvmed149 sdd70 tty84 tty94 tty110 vcs149 vcs150
loop587 nvmed150 loop588 loop589 loop5810 loop5811 loop5812 loop5813 loop5814
nvmed151 sdd71 tty85 tty95 tty111 vcs151 vcs152
loop595 nvmed152 loop596 loop597 loop598 loop599 loop5910 loop5911 loop5912
nvmed153 sdd72 tty86 tty96 tty112 vcs153 vcs154
loop603 nvmed154 loop604 loop605 loop606 loop607 loop608 loop609 loop6010
nvmed155 sdd73 tty87 tty97 tty113 vcs155 vcs156
loop611 nvmed156 loop612 loop613 loop614 loop615 loop616 loop617 loop6110
nvmed157 sdd74 tty88 tty98 tty114 vcs157 vcs158
loop619 nvmed158 loop6110 loop6111 loop6112 loop6113 loop6114 loop6115 loop6116
nvmed159 sdd75 tty89 tty99 tty115 vcs159 vcs160
loop627 nvmed160 loop628 loop629 loop6210 loop6211 loop6212 loop6213 loop6214
nvmed161 sdd76 tty90 tty100 tty116 vcs161 vcs162
loop635 nvmed162 loop636 loop637 loop638 loop639 loop6310 loop6311 loop6312
nvmed163 sdd77 tty91 tty101 tty117 vcs163 vcs164
loop643 nvmed164 loop645 loop646 loop647 loop648 loop649 loop6410 loop6411
nvmed165 sdd78 tty92 tty102 tty118 vcs165 vcs166
loop651 nvmed166 loop652 loop653 loop654 loop655 loop656 loop657 loop6510
nvmed167 sdd79 tty93 tty103 tty119 vcs167 vcs168
loop659 nvmed168 loop660 loop661 loop662 loop663 loop664 loop665 loop6610
nvmed169 sdd80 tty94 tty104 tty120 vcs169 vcs170
loop667 nvmed170 loop668 loop669 loop6610 loop6611 loop6612 loop6613 loop6614
nvmed171 sdd81 tty95 tty105 tty121 vcs171 vcs172
loop675 nvmed172 loop676 loop677 loop678 loop679 loop6710 loop6711 loop6712
nvmed173 sdd82 tty96 tty106 tty122 vcs173 vcs174
loop683 nvmed174 loop684 loop685 loop686 loop687 loop688 loop689 loop6810
nvmed175 sdd83 tty97 tty107 tty123 vcs175 vcs176
loop691 nvmed176 loop692 loop693 loop694 loop695 loop696 loop697 loop6910
nvmed177 sdd84 tty98 tty108 tty124 vcs177 vcs178
loop699 nvmed178 loop6910 loop6911 loop6912 loop6913 loop6914 loop6915 loop6916
nvmed179 sdd85 tty99 tty109 tty125 vcs179 vcs180
loop707 nvmed180 loop708 loop709 loop7010 loop7011 loop7012 loop7013 loop7014
nvmed181 sdd86 tty100 tty110 tty126 vcs181 vcs182
loop715 nvmed182 loop716 loop717 loop718 loop719 loop7110 loop7111 loop7112
nvmed183 sdd87 tty101 tty111 tty127 vcs183 vcs184
loop723 nvmed184 loop725 loop726 loop727 loop728 loop729 loop7210 loop7211
nvmed185 sdd88 tty102 tty112 tty128 vcs185 vcs186
loop731 nvmed186 loop732 loop733 loop734 loop735 loop736 loop737 loop7310
nvmed187 sdd89 tty103 tty113 tty129 vcs187 vcs188
loop739 nvmed188 loop7310 loop7311 loop7312 loop7313 loop7314 loop7315 loop7316
nvmed189 sdd90 tty104 tty114 tty130 vcs189 vcs190
loop747 nvmed190 loop748 loop749 loop7410 loop7411 loop7412 loop7413 loop7414
nvmed191 sdd91 tty105 tty115 tty131 vcs191 vcs192
loop755 nvmed192 loop756 loop757 loop758 loop759 loop7510 loop7511 loop7512
nvmed193 sdd92 tty106 tty116 tty132 vcs193 vcs194
loop763 nvmed194 loop764 loop765 loop766 loop767 loop768 loop769 loop7610
nvmed195 sdd93 tty107 tty117 tty133 vcs195 vcs196
loop771 nvmed196 loop772 loop773 loop774 loop775 loop776 loop777 loop7710
nvmed197 sdd94 tty108 tty118 tty134 vcs197 vcs198
loop779 nvmed198 loop7710 loop7711 loop7712 loop7713 loop7714 loop7715 loop7716
nvmed199 sdd95 tty109 tty119 tty135 vcs199 vcs200
loop787 nvmed200 loop788 loop789 loop7810 loop7811 loop7812 loop7813 loop7814
nvmed201 sdd96 tty110 tty120 tty136 vcs201 vcs202
loop795 nvmed202 loop796 loop797 loop798 loop799 loop7910 loop7911 loop7912
nvmed203 sdd97 tty111 tty121 tty137 vcs203 vcs204
loop799 nvmed204 loop7910 loop7911 loop7912 loop7913 loop7914 loop7915 loop7916
nvmed205 sdd98 tty112 tty122 tty138 vcs205 vcs206
loop803 nvmed206 loop804 loop805 loop806 loop807 loop808 loop809 loop8010
nvmed207 sdd99 tty113 tty123 tty139 vcs207 vcs208
loop811 nvmed208 loop812 loop813 loop814 loop815 loop816 loop817 loop8110
nvmed209 sdd100 tty114 tty124 tty140 vcs209 vcs210
loop819 nvmed210 loop8110 loop8111 loop8112 loop8113 loop8114 loop8115 loop8116
nvmed211 sdd101 tty115 tty125 tty141 vcs211 vcs212
loop827 nvmed212 loop828 loop829 loop8210 loop8211 loop8212 loop8213 loop8214
nvmed213 sdd102 tty116 tty126 tty142 vcs213 vcs214
loop835 nvmed214 loop836 loop837 loop838 loop839 loop8310 loop8311 loop8312
nvmed215 sdd103 tty117 tty127 tty143 vcs215 vcs216
loop843 nvmed216 loop844 loop845 loop846 loop847 loop848 loop849 loop8410
nvmed217 sdd104 tty118 tty128 tty144 vcs217 vcs218
loop851 nvmed218 loop852 loop853 loop854 loop855 loop856 loop857 loop8510
nvmed219 sdd105 tty119 tty129 tty145 vcs219 vcs220
loop859 nvmed220 loop8510 loop8511 loop8512 loop8513 loop8514 loop8515 loop8516
nvmed221 sdd106 tty120 tty130 tty146 vcs221 vcs222
loop867 nvmed222 loop868 loop869 loop8610 loop8611 loop8612 loop8613 loop8614
nvmed223 sdd107 tty121 tty131 tty147 vcs223 vcs224
loop875 nvmed224 loop876 loop877 loop878 loop879 loop8710 loop8711 loop8712
nvmed225 sdd108 tty122 tty132 tty148 vcs225 vcs226
loop883 nvmed226 loop884 loop885 loop886 loop887 loop888 loop889 loop8810
nvmed227 sdd109 tty123 tty133 tty149 vcs227 vcs228
loop891 nvmed228 loop892 loop893 loop894 loop895 loop896 loop897 loop8910
nvmed229 sdd110 tty124 tty134 tty150 vcs229 vcs230
loop899 nvmed230 loop8910 loop8911 loop8912 loop8913 loop8914 loop8915 loop8916
nvmed231 sdd111 tty125 tty135 tty151 vcs231 vcs232
loop907 nvmed232 loop908 loop909 loop9010 loop9011 loop9012 loop9013 loop9014
nvmed233 sdd112 tty126 tty136 tty152 vcs233 vcs234
loop915 nvmed234 loop916 loop917 loop918 loop919 loop9110 loop9111 loop9112
nvmed235 sdd113 tty127 tty137 tty153 vcs235 vcs236
loop923 nvmed236 loop924 loop925 loop926 loop927 loop928 loop929 loop9210
nvmed237 sdd114 tty128 tty138 tty154 vcs237 vcs238
loop931 nvmed238 loop932 loop933 loop934 loop935 loop936 loop937 loop9310
nvmed239 sdd115 tty129 tty139 tty155 vcs239 vcs240
loop939 nvmed240 loop9310 loop9311 loop9312 loop9313 loop9314 loop9315 loop9316
nvmed241 sdd116 tty130 tty140 tty156 vcs241 vcs242
loop947 nvmed242 loop948 loop949 loop9410 loop9411 loop9412 loop9413 loop9414
nvmed243 sdd117 tty131 tty141 tty157 vcs243 vcs244
loop955 nvmed244 loop956 loop957 loop958 loop959 loop9510 loop9511 loop9512
nvmed245 sdd118 tty132 tty142 tty158 vcs245 vcs246
loop963 nvmed246 loop964 loop965 loop966 loop967 loop968 loop969 loop9610
nvmed247 sdd119 tty133 tty143 tty159 vcs247 vcs248
loop971 nvmed248 loop972 loop973 loop974 loop975 loop976 loop977 loop9710
nvmed249 sdd120 tty134 tty144 tty160 vcs249 vcs250
loop979 nvmed250 loop9710 loop9711 loop9712 loop9713 loop9714 loop9715 loop9716
nvmed251 sdd121 tty135 tty145 tty161 vcs251 vcs252
loop987 nvmed252 loop988 loop989 loop9810 loop9811 loop9812 loop9813 loop9814
nvmed253 sdd122 tty136 tty146 tty162 vcs253 vcs254
loop995 nvmed254 loop996 loop997 loop998 loop999 loop9910 loop9911 loop9912
nvmed255 sdd123 tty137 tty147 tty163 vcs255 vcs256
loop1003 nvmed256 loop1004 loop1005 loop10010 loop10011 loop10012 loop10013 loop10014
nvmed257 sdd124 tty138 tty148 tty164 vcs257 vcs258
loop1011 nvmed258 loop1012 loop1013 loop1014 loop1015 loop1016 loop1017 loop10110
nvmed259 sdd125 tty139 tty149 tty165 vcs259 vcs260
loop1019 nvmed260 loop10110 loop10111 loop10112 loop10113 loop10114 loop10115 loop10116
nvmed261 sdd126 tty140 tty150 tty166 vcs261 vcs262
loop1027 nvmed262 loop1028 loop1029 loop10210 loop10211 loop10212 loop10213 loop10214
nvmed263 sdd127 tty141 tty151 tty167 vcs263 vcs264
loop1035 nvmed264 loop1036 loop1037 loop1038 loop1039 loop10310 loop10311 loop10312
nvmed265 sdd128 tty142 tty152 tty168 vcs265 vcs266
loop1043 nvmed266 loop1044 loop1045 loop1046 loop1047 loop1048 loop1049 loop10410
nvmed267 sdd129 tty143 tty153 tty169 vcs267 vcs268
loop1051 nvmed268 loop1052 loop1053 loop1054 loop1055 loop1056 loop1057 loop10510
nvmed269 sdd130 tty144 tty154 tty170 vcs269 vcs270
loop1059 nvmed270 loop10510 loop10511 loop10512 loop10513 loop10514 loop10515 loop10516
nvmed271 sdd131 tty145 tty155 tty171 vcs271 vcs272
loop1067 nvmed272 loop1068 loop1069 loop10610 loop10611 loop10612 loop10613 loop10614
nvmed273 sdd132 tty146 tty156 tty172 vcs273 vcs274
loop1075 nvmed274 loop1076 loop1077 loop1078 loop1079 loop10710 loop10711 loop10712
nvmed275 sdd133 tty147 tty157 tty173 vcs275 vcs276
loop1083 nvmed276 loop1084 loop1085 loop10810 loop10811 loop10812 loop10813 loop10814
nvmed277 sdd134 tty148 tty158 tty174 vcs277 vcs278
loop1091 nvmed278 loop1092 loop1093 loop1094 loop1095 loop1096 loop1097 loop10910
nvmed279 sdd135 tty149 tty159 tty175 vcs279 vcs280
loop1099 nvmed280 loop10910 loop10911 loop10912 loop10913 loop10914 loop10915 loop10916
nvmed281 sdd136 tty150 tty160 tty176 vcs281 vcs282
loop1107 nvmed282 loop1108 loop1109 loop11010 loop11011 loop11012 loop11013 loop11014
nvmed283 sdd137 tty151 tty161 tty177 vcs283 vcs284
loop1115 nvmed284 loop1116 loop1117 loop1118 loop1119 loop11110 loop11111 loop11112
nvmed285 sdd138 tty152 tty162 tty178 vcs285 vcs286
loop1123 nvmed286 loop1124 loop1125 loop11210 loop11211 loop11212 loop11213 loop11214
nvmed287 sdd139 tty153 tty163 tty179 vcs287 vcs288
loop1131 nvmed288 loop1132 loop1133 loop1134 loop1135 loop11310 loop11311 loop11312
nvmed289 sdd140 tty154 tty164 tty180 vcs289 vcs290
loop1139 nvmed290 loop11310 loop11311 loop11312 loop11313 loop11314 loop11315 loop11316
nvmed291 sdd141 tty155 tty165 tty181 vcs291 vcs292
loop1147 nvmed292 loop1148 loop1149 loop11410 loop11411 loop11412 loop11413 loop11414
nvmed293 sdd142 tty156 tty166 tty182 vcs293 vcs294
loop1155 nvmed294 loop1156 loop1157 loop11510 loop11511 loop11512 loop11513 loop11514
nvmed295 sdd143 tty157 tty167 tty183 vcs295 vcs296
loop1163 nvmed296 loop1164 loop1165 loop11610 loop11611 loop11612 loop11613 loop11614
nvmed297 sdd144 tty158 tty168 tty184 vcs297 vcs298
loop1171 nvmed298 loop1172 loop1173 loop11710 loop11711 loop11712 loop11713 loop11714
nvmed299 sdd145 tty159 tty169 tty185 vcs299 vcs300
loop1179 nvmed300 loop11710 loop11711 loop11712 loop11713 loop11714 loop11715 loop11716
nvmed301 sdd146 tty160 tty170 tty186 vcs301 vcs302
loop1187 nvmed302 loop1188 loop1189 loop11810 loop11811 loop11812 loop11813 loop11814
nvmed303 sdd147 tty161 tty171 tty187 vcs303 vcs304
loop1195 nvmed304 loop1196 loop1197 loop11910 loop11911 loop11912 loop11913 loop11914
nvmed305 sdd148 tty162 tty172 tty188 vcs305 vcs306
loop1203 nvmed306 loop1207 loop1208 loop12010 loop12011 loop12012 loop12013 loop12014
nvmed307 sdd149 tty163 tty173 tty189 vcs307 vcs308
loop1211
```

```
reds@reds:~$ sudo nvme id-ctrl -H /dev/nvme0
NVME Identify Controller:
vid      : 0x1edc
ssvid   : 0x1edc
sn       : SSD515T
mn      : Cosmos+ OpenSSD
fr      : TYPE0005
rab     : 0
ieee    : 5cd2e4
cmic    : 0
[2:2] : 0      PCI
[1:1] : 0      Single Controller
[0:0] : 0      Single Port

mdts   : 8
cntllid : 9
ver     : 0
rtd3r   : 0
rtd3e   : 0
oaes    : 0
[8:8] : 0      Namespace Attribute Changed Event Not Supported
ctratt  : 0
[0:0] : 0      128-bit Host Identifier Not Supported

oacs    : 0
[8:8] : 0      Doorbell Buffer Config Not Supported
[7:7] : 0      Virtualization Management Not Supported
[6:6] : 0      NVMe-MI Send and Receive Not Supported
[5:5] : 0      Directives Not Supported
[4:4] : 0      Device Self-test Not Supported
[3:3] : 0      NS Management and Attachment Not Supported
[2:2] : 0      FW Commit and Download Not Supported
[1:1] : 0      Format NVN Not Supported
[0:0] : 0      Security Send and Receive Not Supported

acl     : 3
aerl    : 3
frmw   : 0x3
[4:4] : 0      Firmware Activate Without Reset Not Supported
[3:1] : 0x1    Number of Firmware Slots
[0:0] : 0x1    Firmware Slot 1 Read-Only

lpa     : 0
[2:2] : 0      Extended data for Get Log Page Not Supported
[1:1] : 0      Command Effects Log Page Not Supported
[0:0] : 0      SMART/Health Log Page per NS Not Supported

elpe    : 8
npss    : 0
avsc   : 0
[0:0] : 0      Admin Vendor Specific Commands uses Vendor Specific Format
```

Figure 1.12: Functionality test - NVMe Identify Command - Host PC

```
reds@reds:~$ sudo nvme get-ns-id /dev/nvme0n1
nvme0n1: namespace-id:14740
reds@reds:~$ sudo nvme id-ns /dev/nvme0n1
NVME Identify Namespace 14740:
nsze   : 0x3994
ncap   : 0x3994
nuse   : 0x3994
nsfeat : 0
nlbaf  : 0
flbas  : 0
mc     : 0
dpc    : 0
dps    : 0
nmic   : 0
rescap : 0
fpi    : 0
nawun  : 0
nawupf : 0
nacwu  : 0
nabsn  : 0
nabo   : 0
nabspf : 0
notob  : 0
nvmcap : 0
nguid  : 00000000000000000000000000000000
eui64  : 0000000000000000
lbaf  0 : ms:0  lbadsl:12 rp:0x2 (in use)
```

Figure 1.13: Functionality test - NVMe Namespace List - Host PC

```
File Edit View Search Terminal Help
reds@reds:~$ sudo dd if=../Downloads/test_rand_8MB of=/dev/nvme0n1 bs=4096
[sudo] password for reds:
2000+0 records in
2000+0 records out
8192000 bytes (8.2 MB, 7.8 MiB) copied, 0.0530143 s, 155 MB/s
reds@reds:~$ sudo dd if=/dev/nvme0n1 of=../Desktop/read_test bs=4096 count=1500
1500+0 records in
1500+0 records out
6144000 bytes (6.1 MB, 5.9 MiB) copied, 0.0644825 s, 95.3 MB/s
```

Figure 1.14: Performance test - dd function, write and read - Base Version

the problem was not found.

1.1.5 Optimized Firmware Version

At first, the cause for the poor performance has been attributed to the scheduling of the NAND requests, having taken for granted that the <MemCpy> function was extremely fast. Therefore an optimized version of the firmware was developed in which the converted slice requests are directly executed, not creating any NAND request.

Even if working correctly, performance were just slightly better than the ones

1.1 – Cosmos+ OpenSSD Project

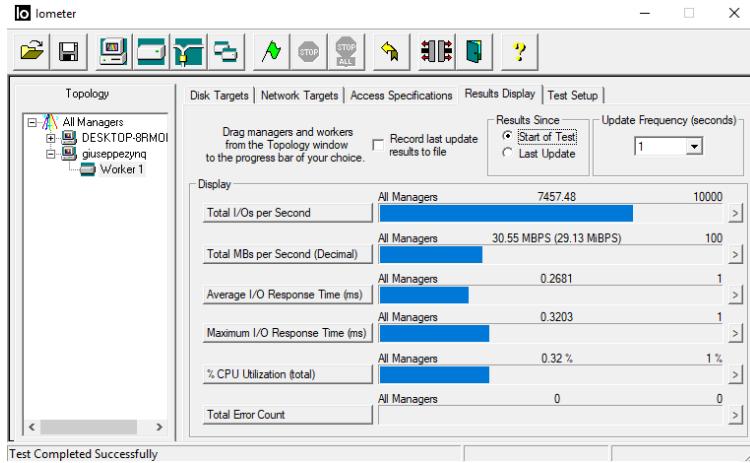


Figure 1.15: Performance test - Iometer Sequential Read transfer size = 4k, block size = 4k - Base Version

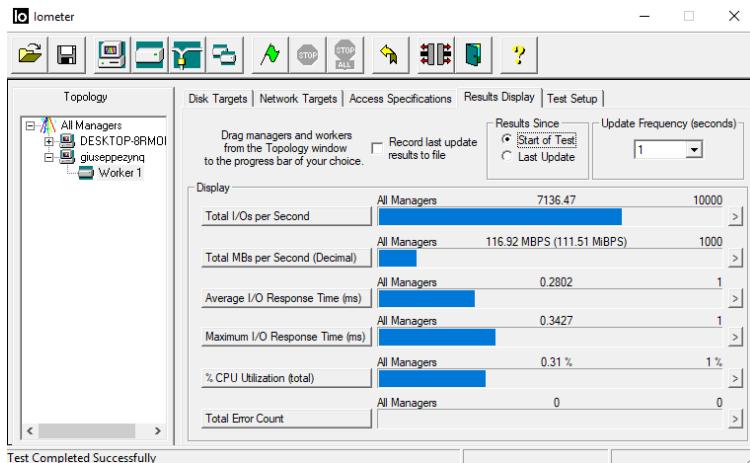


Figure 1.16: Performance test - Iometer Sequential Read transfer size = 16k, block size = 4k - Base Version

of the original firmware, as shown in Figures 1.21,1.22 and 1.23: this helped to discover the real problem, but has lead to no significant improvements. For this reason and to modify the original code the least possible to preserve stability, this version was discarded and no further optimization was carried out.

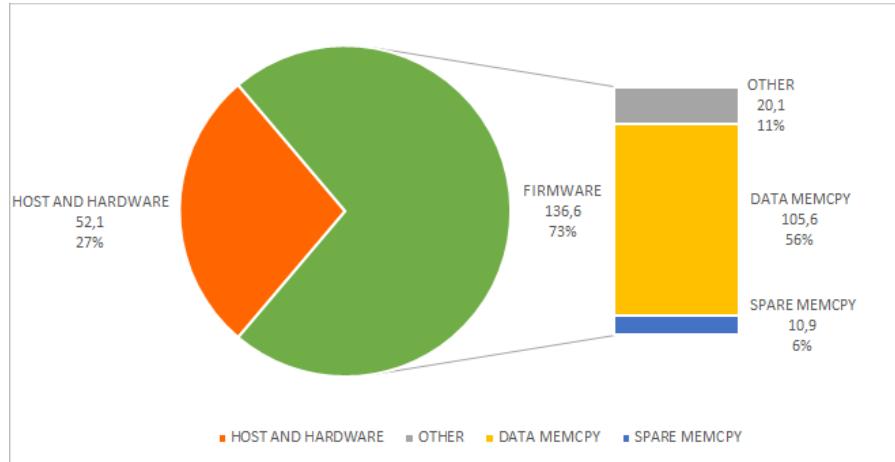


Figure 1.17: Performance test - Read Latency - Base Version

```
reqSlotTag.reqCode: 10; nvmeDmaReqQ.reqCnt: 1, nvmehead: 6, nvmetail: 6
head.autoDmaRx: A9, tail.autoDmaRx: A9, tailIndex: A9, tailAssistIndex: 0; OverFlowCnt: 0
reqSlotTag.reqCode: CB; nvmeDmaReqQ.reqCnt: 0, nvmehead: FFFF, nvmetail: FFFF
reqSlotTag.reqCode: 10; nvmeDmaReqQ.reqCnt: 1, nvmehead: 5, nvmetail: 5
head.autoDmaRx: AA, tail.autoDmaRx: AA, tailIndex: AA, tailAssistIndex: 0; OverFlowCnt: 0
reqSlotTag.reqCode: CB; nvmeDmaReqQ.reqCnt: 0, nvmehead: FFFF, nvmetail: FFFF
reqSlotTag.reqCode: 10; nvmeDmaReqQ.reqCnt: 1, nvmehead: 9, nvmetail: 9
head.autoDmaRx: AA, tail.autoDmaRx: AB, tailIndex: AB, tailAssistIndex: 0; OverFlowCnt: 0
reqSlotTag.reqCode: 10; nvmeDmaReqQ.reqCnt: 1, nvmehead: 9, nvmetail: 9
head.autoDmaRx: AA, tail.autoDmaRx: AB, tailIndex: AB, tailAssistIndex: 0; OverFlowCnt: 0
reqSlotTag.reqCode: 10; nvmeDmaReqQ.reqCnt: 2, nvmehead: 9, nvmetail: A
head.autoDmaRx: AC, tailIndex: AC, tailAssistIndex: 0; OverFlowCnt: 0
```

Figure 1.18: Terminal Error - head.autoDmaRx stuck

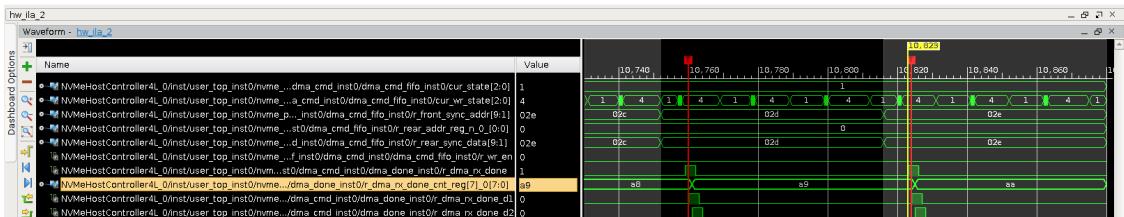


Figure 1.19: Waveform Error - head.autoDmaRx stuck

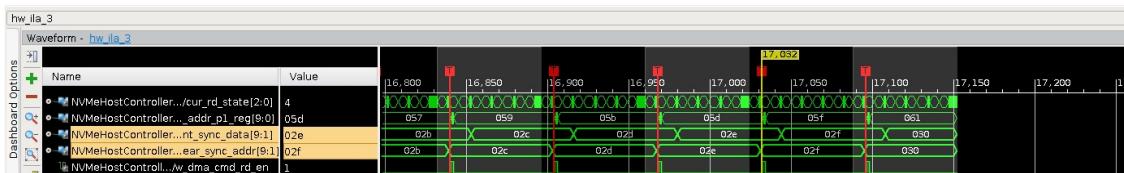
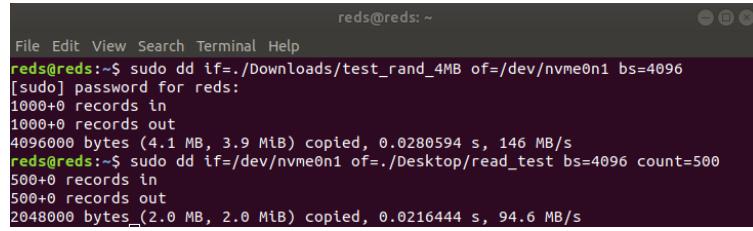


Figure 1.20: Waveform Error - Submitted DMA Request



```

File Edit View Search Terminal Help
reds@reds:~$ sudo dd if=./Downloads/test_rand_4MB of=/dev/nvme0n1 bs=4096
[sudo] password for reds:
1000+0 records in
1000+0 records out
4096000 bytes (4.1 MB, 3.9 MiB) copied, 0.0280594 s, 146 MB/s
reds@reds:~$ sudo dd if=/dev/nvme0n1 of=./Desktop/read_test bs=4096 count=500
500+0 records in
500+0 records out
2048000 bytes (2.0 MB, 2.0 MiB) copied, 0.0216444 s, 94.6 MB/s

```

Figure 1.21: Performance test - dd function, write and read - Optimized Version

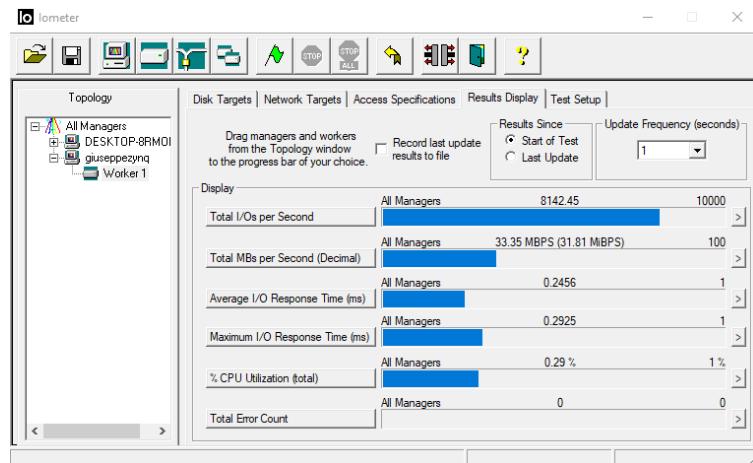


Figure 1.22: Performance test - Iometer Sequential Read transfer size = 4k, block size = 4k - Optimized Version

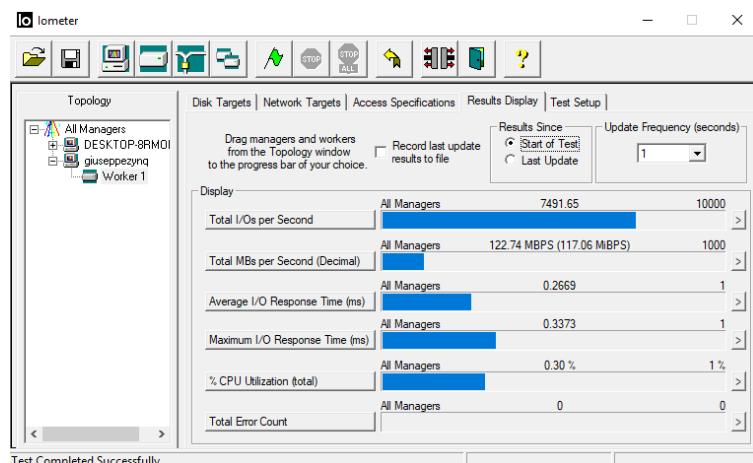


Figure 1.23: Performance test - Iometer Sequential Read transfer size = 16k, block size = 4k - Optimized Version

1.2 Computational Storage Development

The objective is to develop an hardware accelerator consisting of a wrapper that can host different types of acceleration core. However, before starting with the development of the hardware accelerator, it is necessary to modify the project with the addition of the AXI DMA IP, in order to match the general architecture of a FPGA-based Computational Storage Drive 1.24.

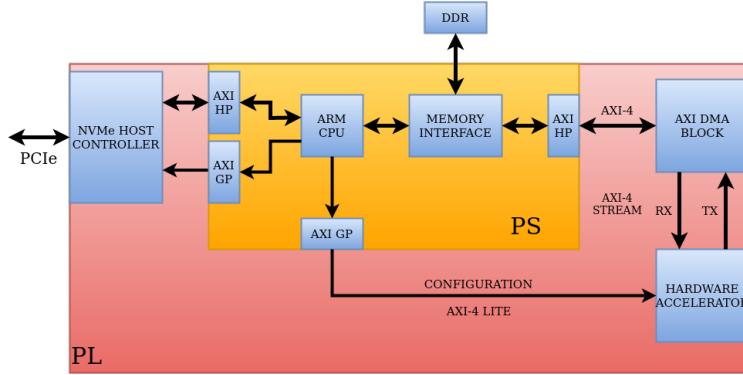


Figure 1.24: General Architecture of a FPGA-based Computational Storage

1.2.1 32-bit AXI DMA

At first, the AXI DMA has been used as a replacement for the MemCpy function, being simply closed on a loop-back. The only firmware modifications concern the configuration of the DMA and, as mentioned above, the substitution of the MemCpy with a DMA transfer.

Some preliminary performance tests are carried out to compare this version to the one with <MemCpy> function.

In Figure 1.26 the <dd> test has been performed: however, due to the small amount of data transferred, the results are not accurate. Instead, as it can be seen in Figures 1.27 to 1.29, despite an increment in the latency, there is a slight increase in both IOPS and bandwidth due to the introduction of the DMA, that reduces the workload of the processing system and favors the increase of throughput.

1.2 – Computational Storage Development

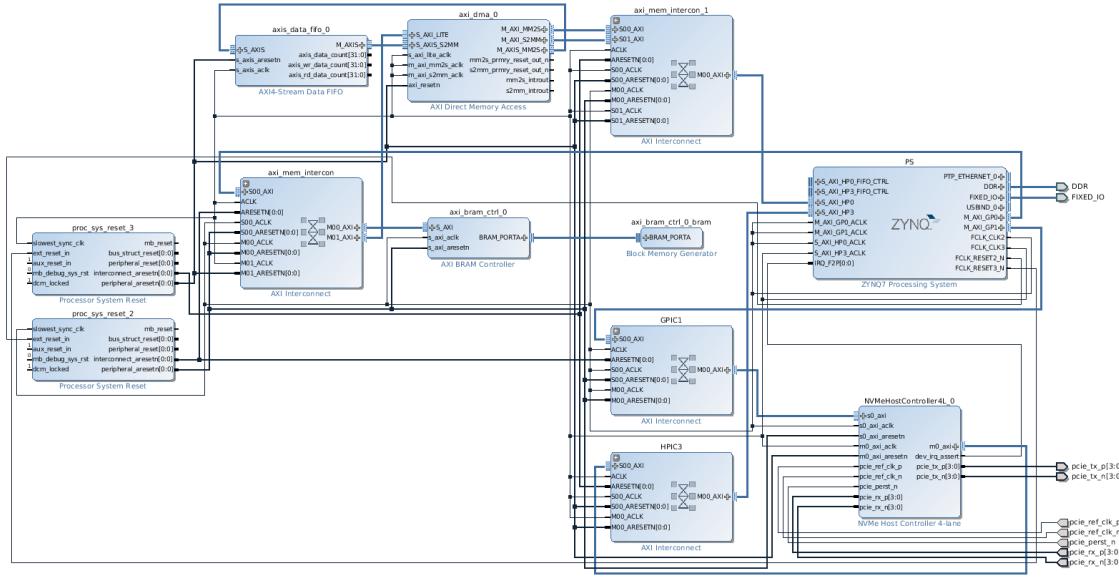


Figure 1.25: 32bit AXI DMA Adaptation of the Cosmos+ OpenSSD project System Design

```
File Edit View Search Terminal Help
redis@redis:~$ sudo dd if=./Downloads/test_rand_2MB of=/dev/nvme0n1 bs=4096
[sudo] password for redis:
500+0 records in
500+0 records out
20480000 bytes (2.0 MB, 2.0 MiB) copied, 0.0251021 s, 81.6 MB/s
redis@redis:~$ sudo dd if=/dev/nvme0n1 of=./Desktop/read_test bs=4096 count=500
500+0 records in
500+0 records out
20480000 bytes (2.0 MB, 2.0 MiB) copied, 0.0236533 s, 86.6 MB/s
```

Figure 1.26: Performance test - dd function, write and read - 32bit AXI DMA Version

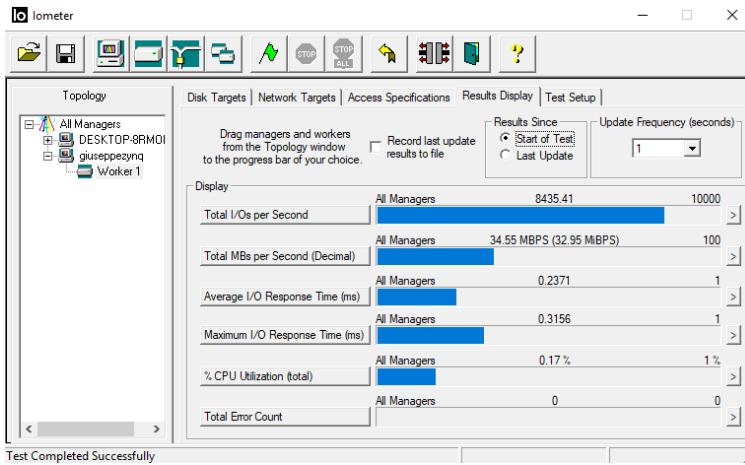


Figure 1.27: Performance test - Iometer Sequential Read transfer size = 4k, block size = 4k - 32bit AXI DMA Adaptation

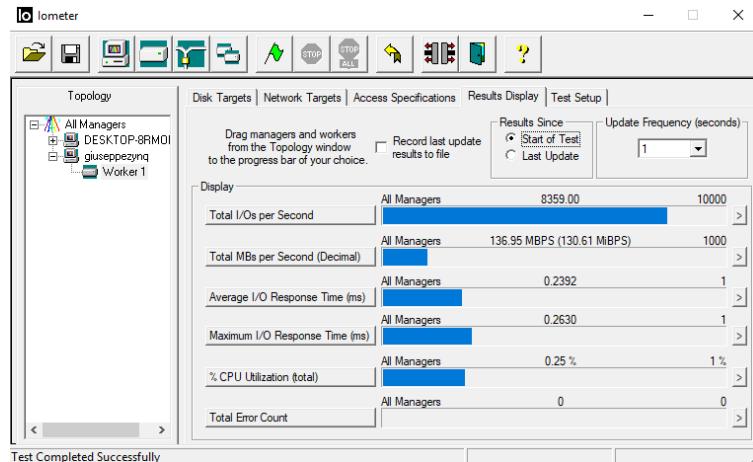


Figure 1.28: Performance test - Iometer Sequential Read transfer size = 16k, block size = 4k - 32bit AXI DMA Adaptation

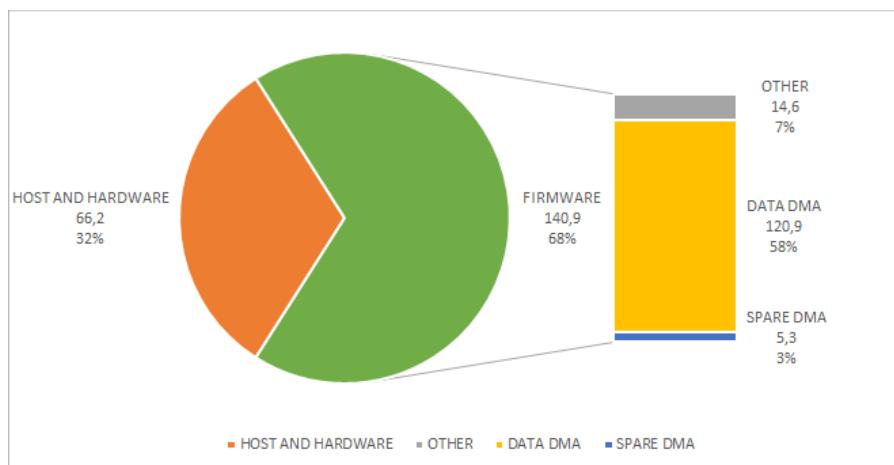


Figure 1.29: Performance test - Read Latency - 32bit AXI DMA Adaptation

1.2.2 First Prototype

The first prototype of hardware accelerator is a simple block which adds the value of a parameter to the data that need to be processed. Its primary goal was to provide a better understanding of the AXI-4 lite and stream protocols and the NVMe Admin Command.

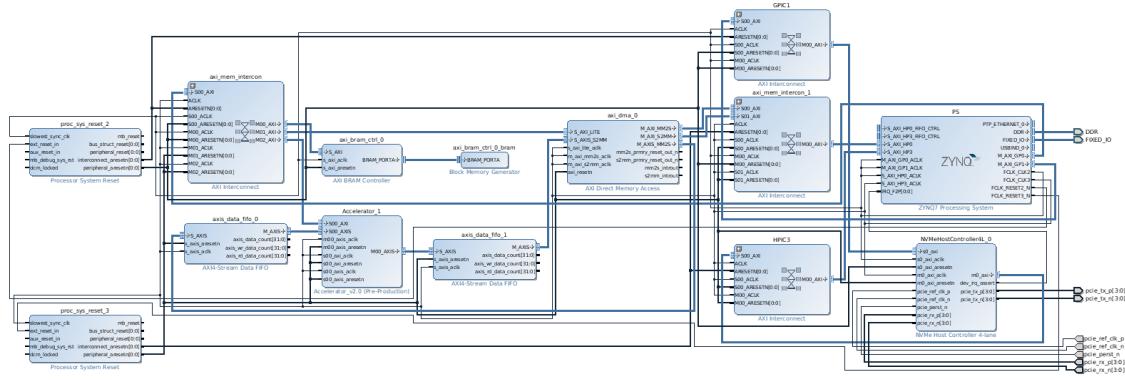


Figure 1.30: System Design of the first version of the Computational Storage based on the Cosmos+ OpenSSD project

As shown in Figure 1.30, the accelerator is enclosed by two Data FIFO, in order to decouple the DMA transfer from the computation.

The designed Finite State Machine (FSM) is shown in Figure 1.31.

The Configuration an Transfer_Configuration state handle respectively the modification and the reading of the configuration and parameter registers through the Set Parameters NVMe Admin Commands. The feature identifiers of the commands, a field that indicates for what feature the attribute are being specified for, are chosen between the group of the vendor specific ones[3], being the registers custom.

On the acceleration branch, the different states are used to take care of all the combination of the AXI-4 stream protocol signals and to recover any lost input data in the occurrence of particular conditions (e.g. output not ready on the last element).

The Accelerator has two configurations: the first is a simple pass-through, necessary for the power-on procedure, while the second is the acceleration configuration that, as previously said, is a simple addition of a parameter to the data. It is possible to change the values of the configuration and parameter registers through the NVMe Set Parameter Command, as shown in Figure 1.32.

The test in Figures 1.33 to 1.36 shows a degradation in performance for this solution due to the insertion of FIFOs and Accelerator block.

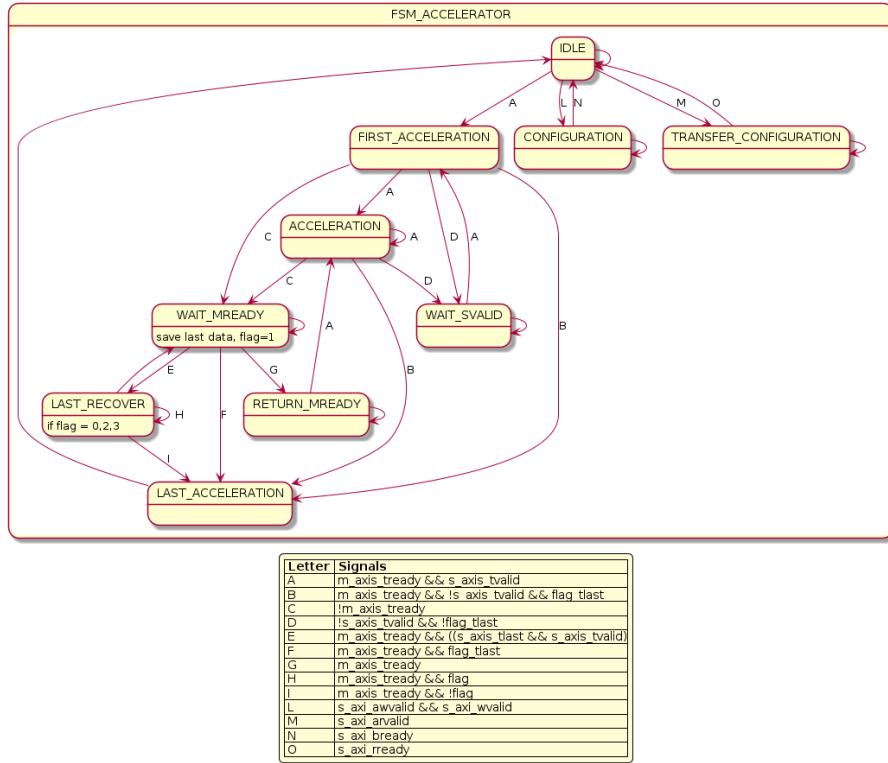


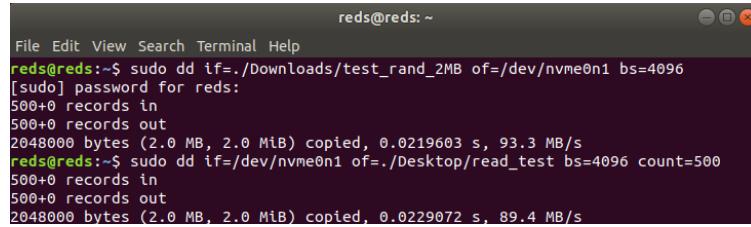
Figure 1.31: FSM of the first version of the Hardware Accelerator

```
reds@reds: ~
File Edit View Search Terminal Help
reds@reds: $ sudo nvme set-feature /dev/nvme0 -f 0xC0 -v 0x1
[sudo] password for reds:
set-feature:c0 (Unknown), value:0x000001
reds@reds: $ sudo nvme set-feature /dev/nvme0 -f 0xC1 -v 0x7
set-feature:c1 (Unknown), value:0x000007
Done Admin Command OPC: C
Done Admin Command OPC: 6
Done Admin Command OPC: 6
Done Admin Command OPC: 6
Set Accelerator Configuration: 1
Set Feature FID:C0
Done Admin Command OPC: 9
Set Accelerator Parameter: 7
Set Feature FID:C1
Done Admin Command OPC: 9
```

Figure 1.32: NVMe Set Parameter Admin Command - Addition of 0x7 to a 32-bit sequence - Host and Developer PCs

1.2.3 128-bit AXI DMA

As it will be treated in the next section, the chosen acceleration core works on 128bit data packet: in order not to add complexity to the accelerator and achieve higher performance, it is necessary to change the AXI DMA bus width from 32 to 128 bits.



```
red@reds: ~
File Edit View Search Terminal Help
[red@reds: ~]$ sudo dd if=~/Downloads/test_rand_2MB of=/dev/nvme0n1 bs=4096
[sudo] password for reds:
500+0 records in
500+0 records out
2048000 bytes (2.0 MB, 2.0 MiB) copied, 0.0219603 s, 93.3 MB/s
[red@reds: ~]$ sudo dd if=/dev/nvme0n1 of=~/Desktop/read_test bs=4096 count=500
500+0 records in
500+0 records out
2048000 bytes (2.0 MB, 2.0 MiB) copied, 0.0229072 s, 89.4 MB/s
```

Figure 1.33: Performance test - dd function, write and read - Computational storage first prototype

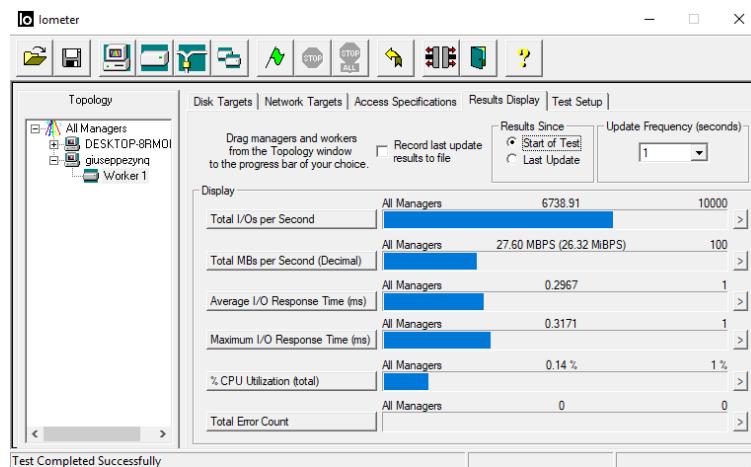


Figure 1.34: Performance test - Iometer Sequential Read transfer size = 4k, block size = 4k - Computational storage first prototype

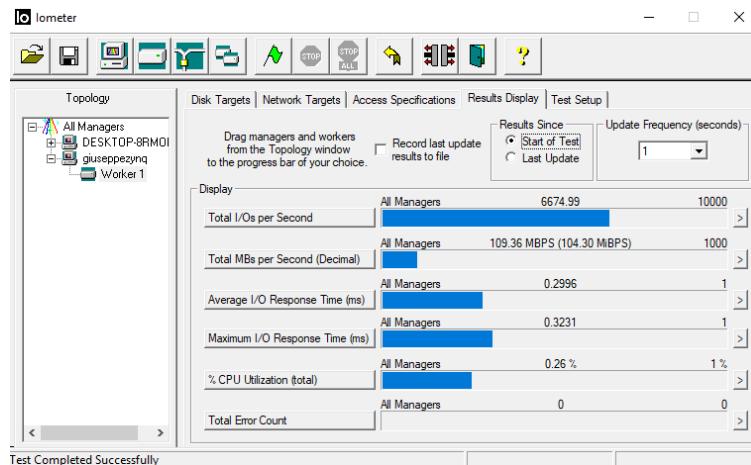


Figure 1.35: Performance test - Iometer Sequential Read transfer size = 16k, block size = 4k - Computational storage first prototype

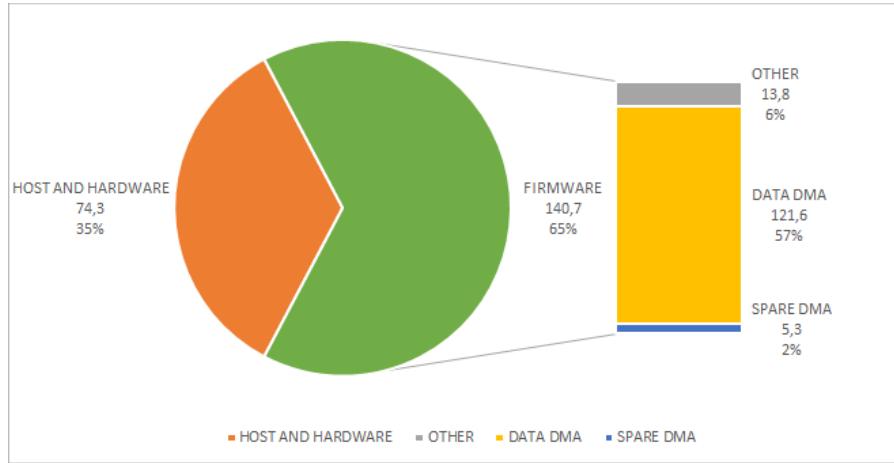


Figure 1.36: Performance test - Read Latency - Computational storage first prototype

Only two firmware modifications has to be carried out in order to perform the 128-bit DMA transfer:

- allocation of a new memory array for the spare data, called <SpaceArray>, to avoid conflicts for consecutive transfers;
- use of an attribute to specify the minimum alignment for the memory arrays, that has to be set to 16 bytes.

As for the previous versions, some preliminary tests are carried out to evaluate the performance of the new configuration.

```
reds@reds: ~
File Edit View Search Terminal Help
reds@reds:~$ sudo dd if=~/Downloads/test_rand_2MB of=/dev/nvme0n1 bs=4096
[sudo] password for reds:
500+0 records in
500+0 records out
2048000 bytes (2.0 MB, 2.0 MiB) copied, 0.0182978 s, 112 MB/s
reds@reds:~$ sudo dd if=/dev/nvme0n1 of=~/Desktop/read_test bs=4096 count=500
500+0 records in
500+0 records out
2048000 bytes (2.0 MB, 2.0 MiB) copied, 0.0198503 s, 103 MB/s
```

Figure 1.37: Performance test - dd function, write and read - 128bit AXI DMA Version

The achieved performance, in terms of bandwidth, IOPS and latency, for the 128-bit AXI DMA version are better than those of the 32-bit one thanks to the higher speed of the data transfer from and to the PS. However, as seen in section 1.1.1, the PS can provide only a 64-bit AXI interface, limiting the possible performance increase.

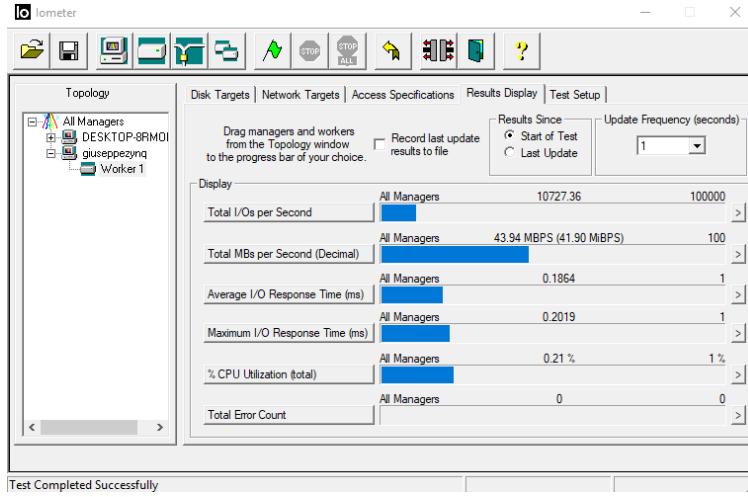


Figure 1.38: Performance test - Iometer Sequential Read transfer size = 4k, block size = 4k - 128bit AXI DMA Adaptation

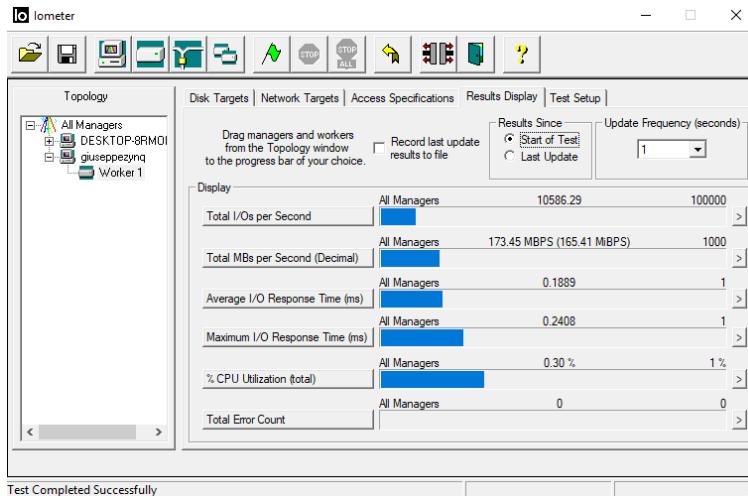


Figure 1.39: Performance test - Iometer Sequential Read transfer size = 16k, block size = 4k - 128bit AXI DMA Adaptation

1.2.4 Second Prototype

The first prototype is not very suitable as general wrapper because the AXI-4 stream protocol makes the management of cores with different timing difficult. Therefore, the previous external FIFOs are then included in the hardware of this second prototype, in order to separate the input and the output AXI-4 stream signals, as shown in Figure 1.41.

In this second prototype, there are two FSMs, as shown in Figure 1.42: the modification and the reading of the registers through NVMe Admin Commands is

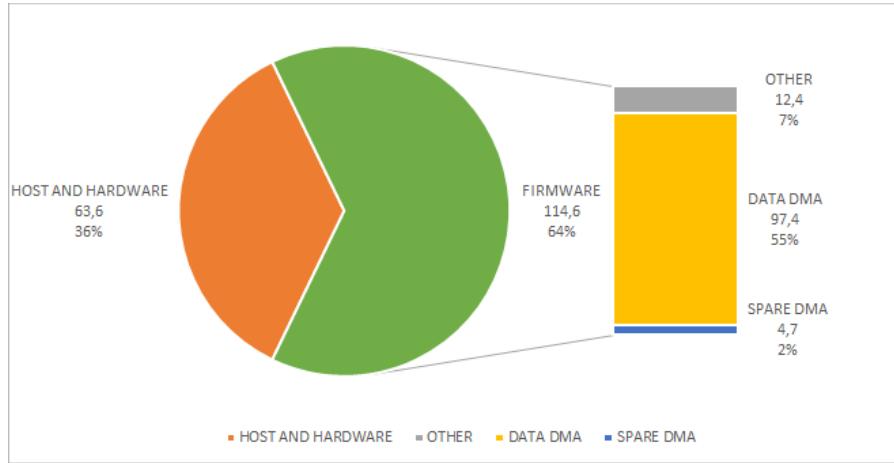


Figure 1.40: Performance test - Read Latency - 128bit AXI DMA Adaptation

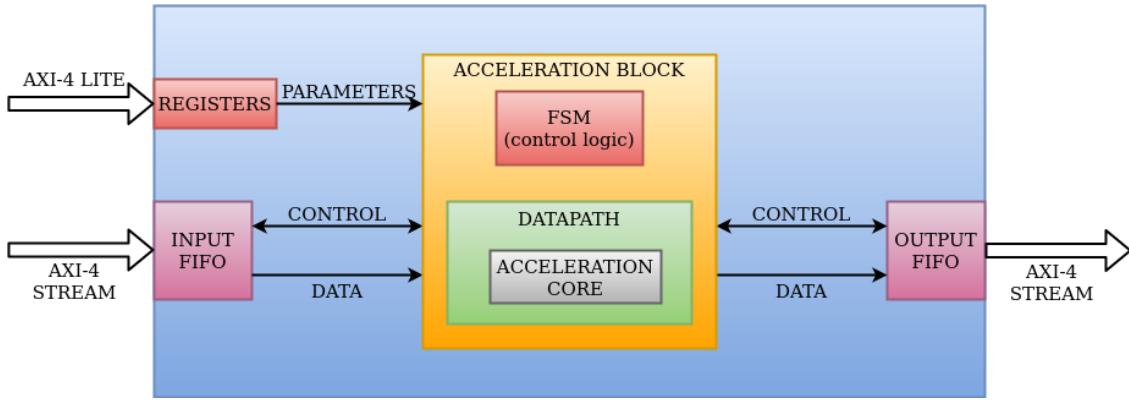


Figure 1.41: General diagram of the Hardware Accelerator

now divided from the acceleration branch.

The acceleration branch is simpler than in the first prototype, due to the acceleration process being independent from the AXI-4 stream protocol. The WAIT_PIPE state is used to wait for the core to be ready, in the case its output is not immediately available.

AES Core and CTR Configuration

In order to verify the correct functioning of the block, it was necessary to choose the acceleration function and, consequently, the core to be inserted.

The sought acceleration function has to be:

- widespread and standard;
- format-preserving: for construction limits (DMA transfer) the output data

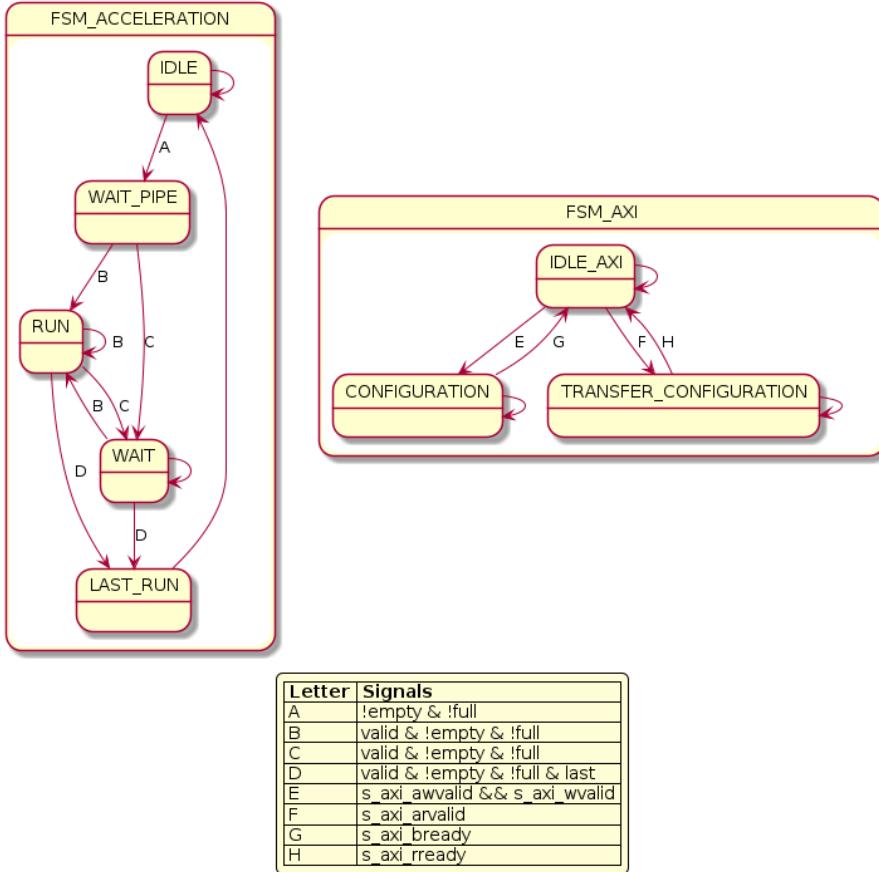


Figure 1.42: FSMs of the second version of the Hardware Accelerator

format has to be the same of the input one;

- used with a data stream: to achieve high performance.

Therefore, the chosen core is an encryption one based on the Advanced Encryption Standard (AES)[4], called "tiny_aes". The AES encryption algorithm, that is a specific implementation of the Rijndael one, has been adopted and standardized by the National Institute of Standards and Technology (NIST) and the US FIPS PUB in 2001 and is accepted all over the world. The AES is a symmetric encryption algorithm, using a single key for both encryption and decryption. It is a format-preserving encryption algorithm that works on a single fixed-length data block of 128-bit. On the other hand, the key size can be 128,192 or 256 bit.

The encryption process consists of different rounds of transformations, which number varies depending on the length of the key.

Five modes of operation were defined[5]:

- Electronic Code Book (ECB) mode: the simplest mode and, for this reason,

generally not recommended. All the input data blocks, called plain text, are encrypted with same key, allowing parallel encryption but resulting in low level of security;

- Cipher Block Chaining (CBC) mode: the input block is xored with an initialization vector (IV) and then encrypted. The resulting cipher text is used as IV for the next block;
- Cipher FeedBack (CFB) mode: the key and the IV are the input of the encryption block, which result is xored with the plain text. As in the CBC, the resulting cipher text is used as IV for the next block;
- Output FeedBack (OFB) mode: as in the CFB, the key and the IV are the input of the encryption block, which result is xored with the plain text to give the cipher text. However the IV for the next block is given by the result of the encryption block;
- Counter (CTR) mode: in this mode the IV is a counter. As in the CFB, the key and the IV are the input of the encryption block, which result is xored with the plain text to give the cipher text. However the IV for the next block is given by the incremented counter.

Apart from the ECB, each mode has advantages and disadvantages: the choice on which to use depends on the application. Given the requirements for the accelerator, the best choice is the CTR mode: in fact in this mode the encryption and decryption can be prepared in advance and the parallel computing is supported, providing in this way the possibility to achieve high speed while ensuring security. Then the hardware accelerator is modified according to the scheme shown in Figure 1.43, in which each block is equivalent of a pipeline stage.

It is fundamental that a data block uses the same value of the counter in both encryption and decryption. Otherwise, the operation is not completed correctly.

As described in the project specification[4], there are three cores available, one for each key size: due to the limit in FPGA resources, only the AES-128 and AES-256 are included in the hardware accelerator.

Functionality Tests

To verify the correct behavior of the developed accelerator, the NIST provides test vectors[5] for both encryption and decryption operations.

By construction of the cores, input and output of the core have to be reversed in order to be processed. This occurs for the opposite orientation of the software and hardware vectors.

Accounting NIST test vectors to be in a reversed state, only the output of the core is inverted: the key has to be inserted as it is. As well, to verify that results match, both plain-text and cipher-text have to be reversed.

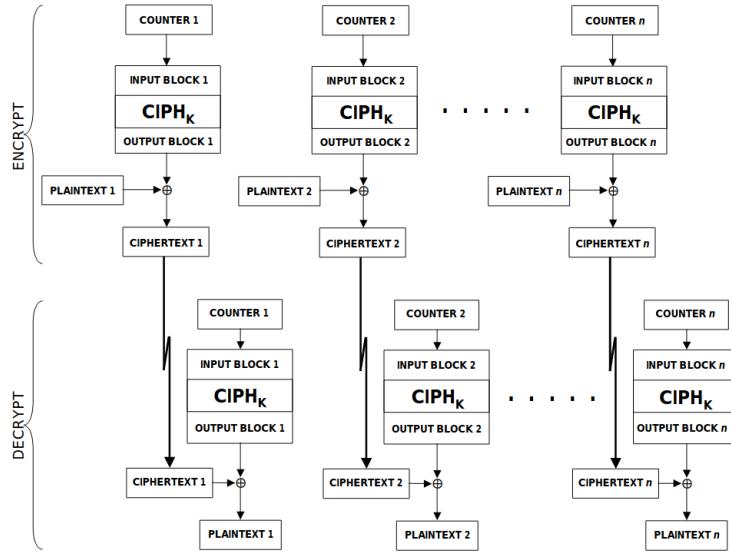


Figure 1.43: CTR Mode - NIST, Recommendation for Block Cipher Modes of Operation: Methods and Techniques

The test are performed in three different ways:

- Module verilog test-bench;
- Module execution;
- General execution.

As shown in Figures the first vectors of the test-bench waveform are, respectively, plain-text blocks, initial counter and key from NIST test vectors.

The plain-text is reversed in the `<data_in_pipe>` vector to ensure the correct behavior: the match of results can be verified by the vector `<reverse_data>`, inversion of the accelerator output.

For the module execution test, the comparison between the obtained cipher-text and NIST test vectors is directly executed by the software. In this type of test the configuration still takes place with direct writing of configuration and parameter registers.

Figures 1.48 and 1.51 shows the messages printed by the software, as a consequence of the correct completion of the operations, for both configurations.

In Figures 1.49, 1.50, 1.52 and 1.53 instead, the complete execution waveforms for the parameters configuration and the encryption phase only are shown.

As mentioned previously, it is possible to notice the difference between the time passed in the state WAIT_PIPE (code 2) in the two encryption waveforms: the process takes more rounds of transformations to complete the operations in the case of the AES-256 core rather than the AES-128 one.

Computational Storage Project

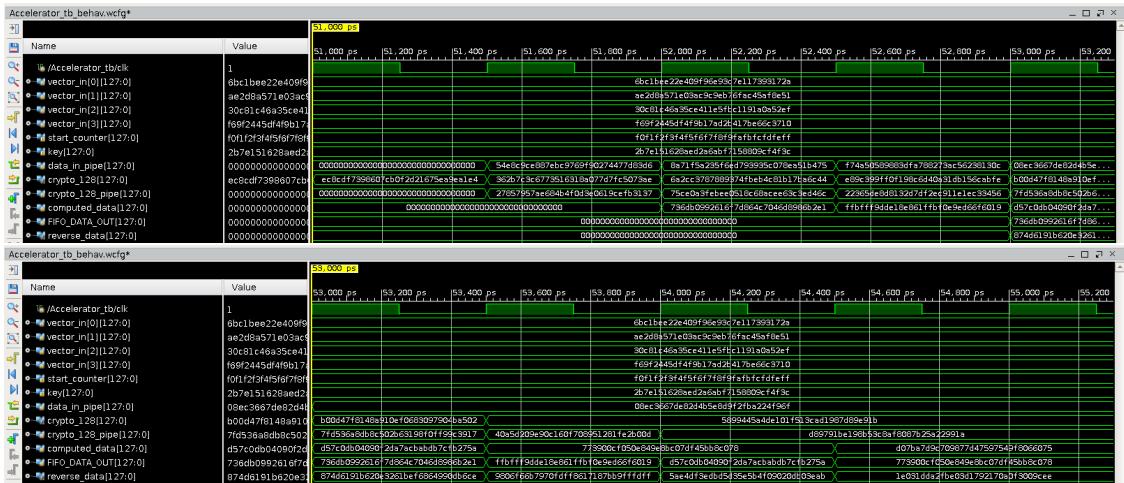


Figure 1.44: Verilog Test-bench - AES-128 Encryption



Figure 1.45: Verilog Test-bench - AES-128 Decryption

Before performing the general test, is necessary to define in the firmware the Feature Identifiers of the Set Feature command. Their assignation to the configuration and parameters registers is shown in Table 1.1.

Finally, it is possible to perform the general test. Figures 1.54 and 1.55 show, from both host and developer PC sides, the configuration phase and write operation to send the test vectors: <test_128.bin> is a binary file in which the NIST vectors have been reversed. The printed output, which are the hexadecimal values of the byte saved in the MemSpace array, are equivalent to test vectors reversed.

The write operations are always referred to a page (16kB): therefore, as it can be seen in Figure 1.31, the amount of computed data is much larger than the 64 bytes of binary file.

1.2 – Computational Storage Development

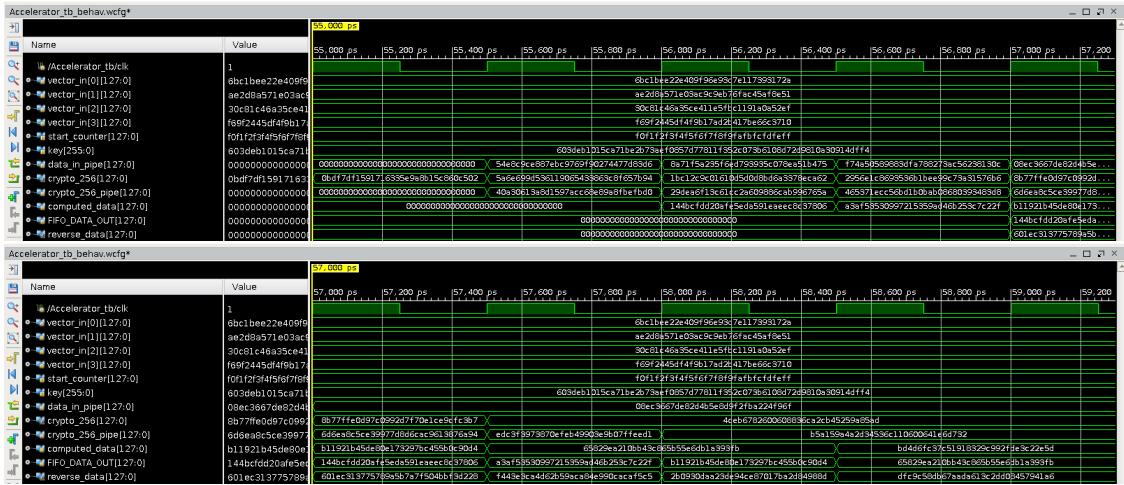


Figure 1.46: Verilog Test-bench - AES-256 Encryption



Figure 1.47: Verilog Test-bench - AES-256 Decryption

```
Terminal ready
*![!] MMU has been enabled.

Hello COSMOS+ OpenSSD !!!
Configuration - Press 'X' to start
Configuration: 02; Key 128bit = 2B7E1516-28AED2A6-ABF71588-9CF4F3C; IV: F0F1F2F3-F4F5F6F7-F8F9FAB-FCDFEFF
Encryption-Decryption - Press 'X' to start
transfer Dev-DMA successful
transfer DMA-Dev successful
Encryption completed successfully
transfer Dev-DMA successful
transfer DMA-Dev successful
Decryption completed successfully
done
```

Figure 1.48: Module execution - AES-128 Terminal

Same operations are performed for the AES-256 core in Figures 1.57 to 1.59.

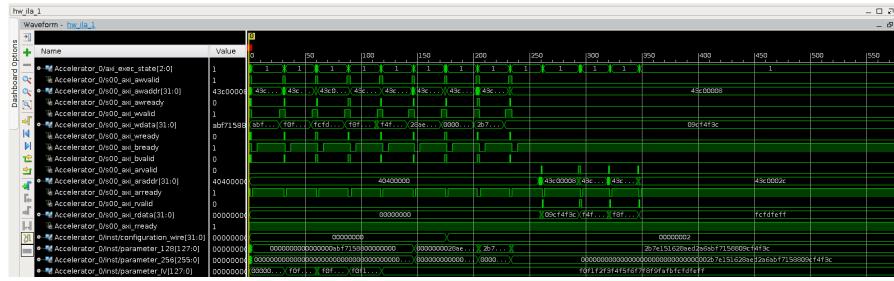


Figure 1.49: Module execution - AES-128 Configuration

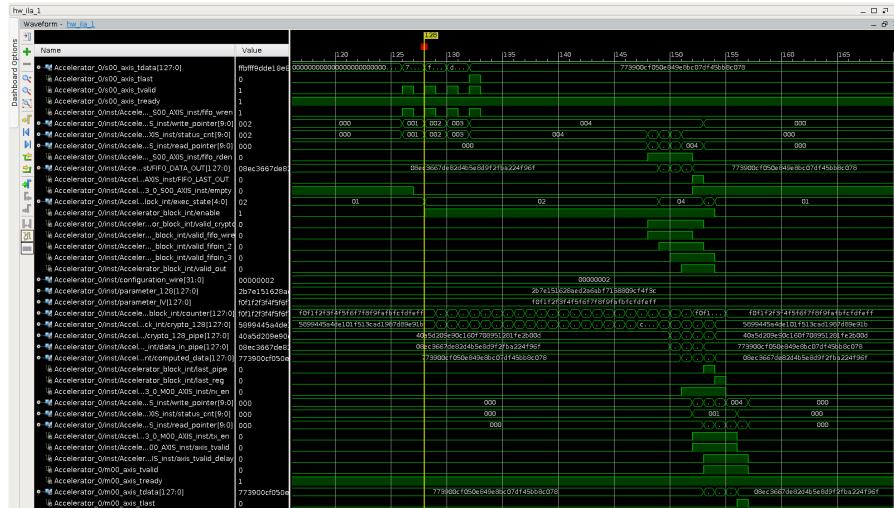


Figure 1.50: Module execution - AES-128 Encryption

```
[root] MMU has been enabled.  
[Hello COSMOS+ OpensSD !!!  
Configuration - Press 'X' to start  
Configuration - 0810000000000000 = -003DEB19-15CA71BE-2B73AEEF-857D7781|F1352C07-3B6108D7-2D9810A3-9140FF4; IV: - 252579085-F4F5F0F7-F8F9FAFB-FCFDFF  
Encryption description - Press 'X' to start  
transfer DMA-successful  
transfer DMA-dev successful  
transfer DMA-addr successfully  
transfer DMA-ctrl successful  
transfer DMA-Dev successful  
Encryption completed successfully  
done
```

Figure 1.51: Module execution - AES-256 Terminal

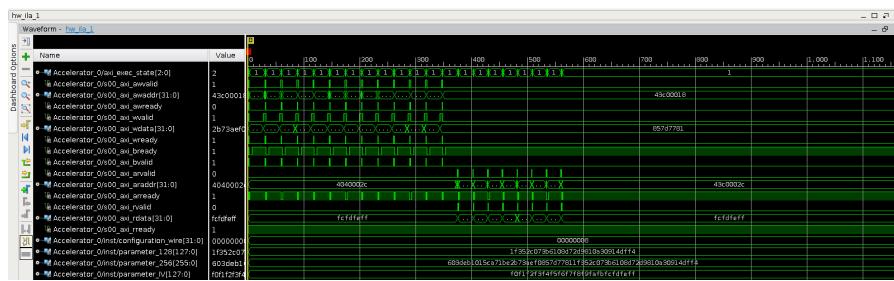


Figure 1.52: Module execution - AES-256 Configuration

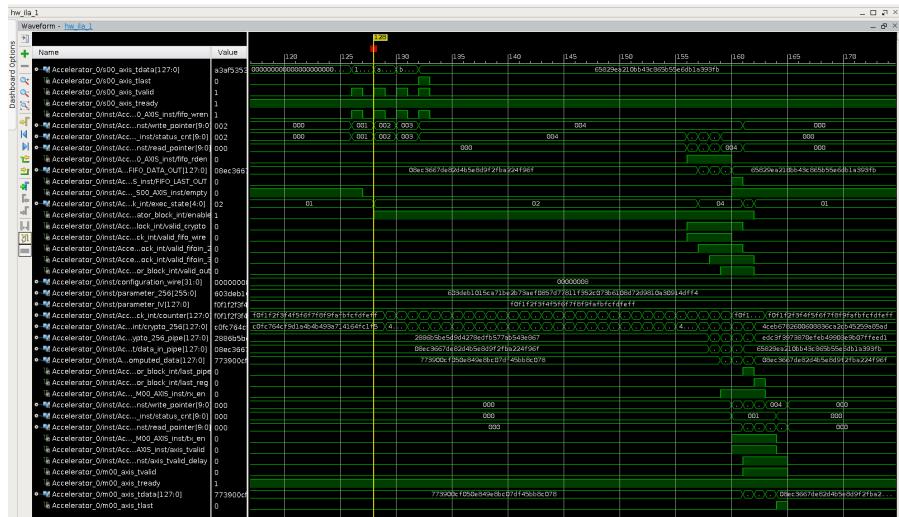


Figure 1.53: Module execution - AES256 Encryption

Feature Identifier	Register Name	Notes
0xC0	Configuration Register	0 = Pass through; 1 = Adder; 2 = AES-128; 8 = AES-256
0xC1	Parameter 1 Register	LSW of the key vector
0xC2	Parameter 2 Register	
0xC3	Parameter 3 Register	
0xC4	Parameter 4 Register	MSW of the 128-bit key vector
0xC5	Parameter 5 Register	
0xC6	Parameter 6 Register	
0xC7	Parameter 7 Register	
0xC8	Parameter 8 Register	MSW of the 256-bit key vector
0xCA	Parameter A Register	LSW of the IV vector
0xCB	Parameter B Register	
0xCC	Parameter C Register	
0xCD	Parameter D Register	MSW of the IV vector

Table 1.1: Set Parameter - Feature Identifiers Assigntion

```

reds@reds: ~
File Edit View Search Terminal Help
[sudo] password for reds:
set-feature:c0 (Unknown), value:0x000002
reds@reds: ~$ sudo nvme set-feature /dev/nvme0 -f 0xC1 -v 0x9cf4f3c
set-feature:c1 (Unknown), value:0x9cf4f3c
reds@reds: ~$ sudo nvme set-feature /dev/nvme0 -f 0xC2 -v 0xabf71588
set-feature:c2 (Unknown), value:0xabf71588
reds@reds: ~$ sudo nvme set-feature /dev/nvme0 -f 0xC3 -v 0x28aed2a6
set-feature:c3 (Unknown), value:0x28aed2a6
reds@reds: ~$ sudo nvme set-feature /dev/nvme0 -f 0xC4 -v 0x2b7e1516
set-feature:c4 (Unknown), value:0x2b7e1516
reds@reds: ~$ sudo nvme set-feature /dev/nvme0 -f 0xCA -v 0xfcfdfeff
set-feature:ca (Unknown), value:0xfcfdfeff
reds@reds: ~$ sudo nvme set-feature /dev/nvme0 -f 0xCB -v 0xf8f9fafb
set-feature:cb (Unknown), value:0xf8f9fafb
reds@reds: ~$ sudo nvme set-feature /dev/nvme0 -f 0xCC -v 0xf4ff5f6f7
set-feature:cc (Unknown), value:0xf4ff5f6f7
reds@reds: ~$ sudo nvme set-feature /dev/nvme0 -f 0xCD -v 0xf0f1f2f3
set-feature:cd (Unknown), value:0xf0f1f2f3
reds@reds: ~$ sudo dd if=/Desktop/test_128.bin of=/dev/nvme0n1 bs=4096
0+1 records in
0+1 records out
64 bytes copied, 0.000307475 s, 208 kB/s

```

Figure 1.54: General Execution - AES-128 Host PC Terminal

```

Set Accelerator Configuration: 2
Done Admin Command OPC: 9
Set Accelerator Parameter 1: 9CF4F3C
Set Feature FID1C
Set Accelerator Parameter 2: ABF71588
Set Feature FID:C2
Set Feature FID:C3
Set Accelerator Parameter 3: 28AE02A6
Set Feature FID:C4
Done Admin Command OPC: 9
Done Admin Command OPC: 4: 2B7E1516
Set Feature FID:C5
Set Admin Command OPC: 9
Set Feature FID:C6
Done Admin Command OPC: A: FCFFDEFF
Set Feature FID:C7
Set Admin Command OPC: 9
Set Feature FID:C8
Done Admin Command OPC: B: FBF9FAFB
Set Feature FID:C9
Set Admin Command OPC: 9
Set Feature FID:C10
Set Admin Command OPC: D: F0F1F2F3
Set Feature FID:C11
Set Admin Command OPC: 9
Done Admin Command OPC: 6
Encryption completed successfully
Mem[16384]:00 -Mem[16387]:7D -Mem[16387]:47 -Mem[16388]:74 -Mem[16389]:02 -Mem[16391]:97 -Mem[16393]:BC -Mem[16394]:7E -Mem[16395]:88
Mem[16397]:75 -Mem[16397]:00 -Mem[16398]:E8 -Mem[16399]:54 -Mem[16400]:75 -Mem[16401]:04 -Mem[16402]:51 -Mem[16403]:EA -Mem[16404]:78 -Mem[16405]:C0 -Mem[16406]:35 -Mem[16407]:39 -Mem[16408]:79 -Mem[16409]:E6 -Mem[16410]:F6 -Mem[16411]:35 -Mem[16412]:A2 -Mem[16413]:F5 -Mem[16414]:71 -Mem[16415]:8A -Mem[16416]:00 -Mem[16417]:00 -Mem[16418]:00 -Mem[16419]:00 -Mem[16420]:CS -Mem[16421]:3A -Mem[16422]:27 -Mem[16423]:88 -Mem[16424]:A7 -Mem[16425]:DF -Mem[16426]:83 -Mem[16427]:98 -Mem[16428]:5B -Mem[16429]:50 -Mem[16430]:4A -Mem[16431]:F7 -Mem[16432]:0F -Mem[16433]:F9 -Mem[16434]:24 -Mem[16435]:A2 -Mem[16436]:FB -Mem[16437]:F2 -Mem[16438]:D9 -Mem[16439]:E8 -Mem[16440]:B5 -Mem[16441]:04 -Mem[16442]:82 -Mem[16443]:DE -Mem[16444]:67 -Mem[16445]:36 -Mem[16446]:EC -Mem[16447]:0B -

```

Figure 1.55: General Execution - AES-128 Developer PC Terminal

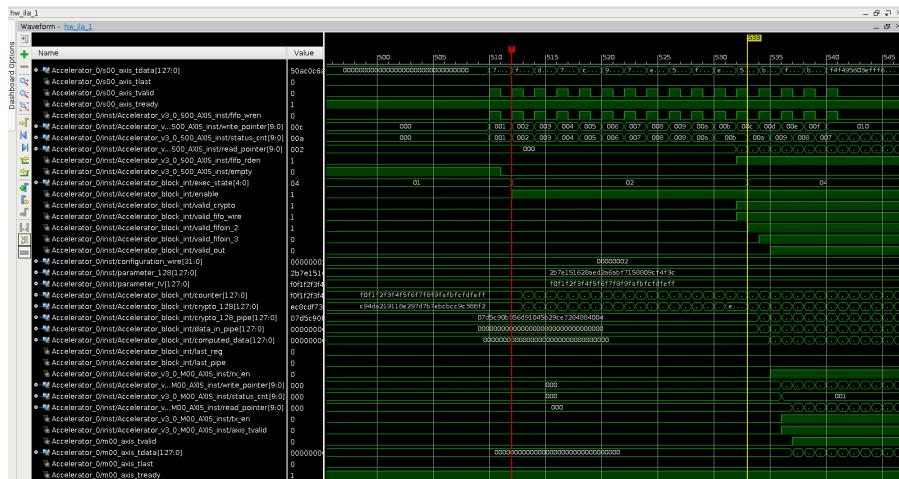


Figure 1.56: General Execution - AES-128 Encryption Waveform

1.2 – Computational Storage Development

```

reds@reds:~$ sudo nvme set-feature /dev/nvme0 -f 0xC0 -v 0x8
(sudo) password for reds:
set-feature:c0 (Unknown), value:0x000008
reds@reds:~$ sudo nvme set-feature /dev/nvme0 -f 0xC1 -v 0x0914dff4
set-feature:c1 (Unknown), value:0x914dff4
reds@reds:~$ sudo nvme set-feature /dev/nvme0 -f 0xC2 -v 0x2d9810a3
set-feature:c2 (Unknown), value:0x2d9810a3
reds@reds:~$ sudo nvme set-feature /dev/nvme0 -f 0xC3 -v 0xb3b6108d7
set-feature:c3 (Unknown), value:0xb3b6108d7
reds@reds:~$ sudo nvme set-feature /dev/nvme0 -f 0xC4 -v 0x1f352c07
set-feature:c4 (Unknown), value:0x1f352c07
reds@reds:~$ sudo nvme set-feature /dev/nvme0 -f 0xC5 -v 0x857d7781
set-feature:c5 (Unknown), value:0x857d7781
reds@reds:~$ sudo nvme set-feature /dev/nvme0 -f 0xC6 -v 0x2b73aaef0
set-feature:c6 (Unknown), value:0x2b73aaef0
reds@reds:~$ sudo nvme set-feature /dev/nvme0 -f 0xC7 -v 0x15ca71be
set-feature:c7 (Unknown), value:0x15ca71be
reds@reds:~$ sudo nvme set-feature /dev/nvme0 -f 0xC8 -v 0x603debf0
set-feature:c8 (Unknown), value:0x603debf0
reds@reds:~$ sudo nvme set-feature /dev/nvme0 -f 0xCA -v 0xfcfdfeff
set-feature:ca (Unknown), value:0xfcfdfeff
reds@reds:~$ sudo dd if=./Desktop/test_256.bin of=/dev/nvme0n1 bs=4096
41+ records in
41+ records out
64 bytes copied, 0.000245735 s., 260 kB/s

```

Figure 1.57: General Execution - AES-256 Host PC Terminal

```

Set Accelerator Configuration: 8
Done Admin Command OPC: 9
Set Accelerator Parameter 1: 9140FF4
Set Feature FID1C
Done Admin Command OPC: 9
Set Accelerator Parameter 2: 2D9810A3
Set Feature FID1C
Done Admin Command OPC: 9
Set Accelerator Parameter 3: 3B6108D7
Set Feature FID1C
Done Admin Command OPC: 9
Set Accelerator Parameter 4: 1F352C07
Set Feature FID1C
Done Admin Command OPC: 9
Set Accelerator Parameter 5: 857D7781
Set Feature FID1C
Done Admin Command OPC: 9
Set Accelerator Parameter 6: 2B73AAEF0
Set Feature FID1C
Done Admin Command OPC: 9
Set Accelerator Parameter 7: 15CA71BE
Set Feature FID1C
Done Admin Command OPC: 9
Set Accelerator Parameter 8: 603DEBF0
Set Feature FID1C
Done Admin Command OPC: 9
Set Accelerator Parameter A: FCFCFDFF
Set Feature FID1C
Done Admin Command OPC: 9
Set Accelerator Parameter B: F8F9FAFB
Set Feature FID1C
Done Admin Command OPC: 9
Set Accelerator Parameter C: F4F5F6F7
Set Feature FID1C
Done Admin Command OPC: 9
Set Accelerator Parameter D: F0F1F2F3
Set Feature FID1C
Done Admin Command OPC: 9
Done Admin Command OPC: 6
Encryption completed successfully
Done[16384]:0[16385]:183 -Mem[16386]:70 -Mem[16387]:74 -Mem[16389]:02 -Mem[16390]:F9 -Mem[16391]:69 -Mem[16392]:97 -Mem[16393]:BC -Mem[16394]:7E -Mem[16395]:88
-Mem[16396]:CE -Mem[16397]:C0 -Mem[16398]:E8 -Mem[16399]:54 -Mem[16400]:75 -Mem[16401]:84 -Mem[16402]:1 -Mem[16403]:EA -Mem[16404]:78 -Mem[16405]:C0 -Mem[16406]:35 -Mem[16407]:39 -Mem[16408]:79 -Mem[16409]:ED -Mem[16410]:F6 -Mem[16411]:35
-Mem[16412]:44 -Mem[16413]:58 -Mem[16414]:62 -Mem[16415]:73 -Mem[16416]:87 -Mem[16417]:13 -Mem[16418]:38 -Mem[16419]:02 -Mem[16420]:C5 -Mem[16421]:27 -Mem[16422]:27 -Mem[16423]:88 -Mem[16424]:A7 -Mem[16425]:D8 -Mem[16426]:83 -Mem[16427]:98
-Mem[16428]:58 -Mem[16429]:59 -Mem[16430]:4A -Mem[16431]:F7 -Mem[16432]:58 -Mem[16433]:62 -Mem[16434]:73 -Mem[16435]:88 -Mem[16436]:FB -Mem[16437]:F2 -Mem[16438]:D9 -Mem[16439]:E8 -Mem[16440]:B5 -Mem[16441]:D4 -Mem[16442]:82 -Mem[16443]:DE
-Mem[16444]:67 -Mem[16445]:36 -Mem[16446]:EC -Mem[16447]:98 -
Done Admin Command OPC: 6

```

Figure 1.58: General Execution - AES-256 Developer PC Terminal

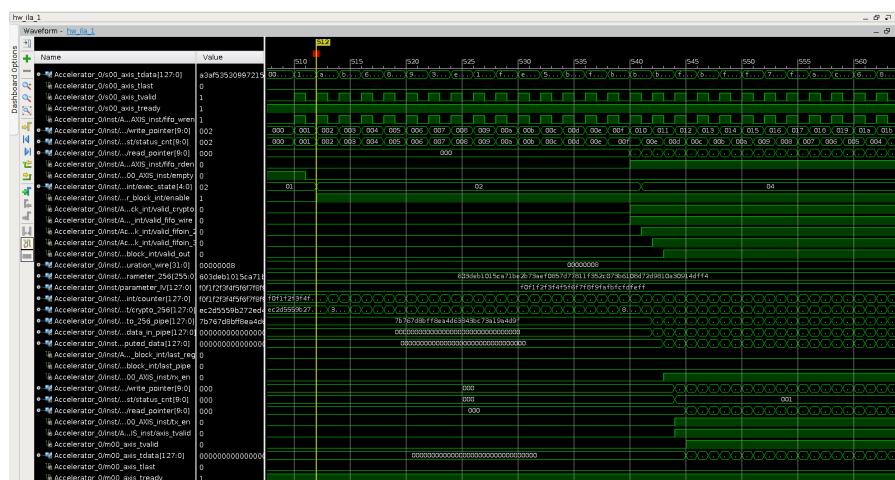
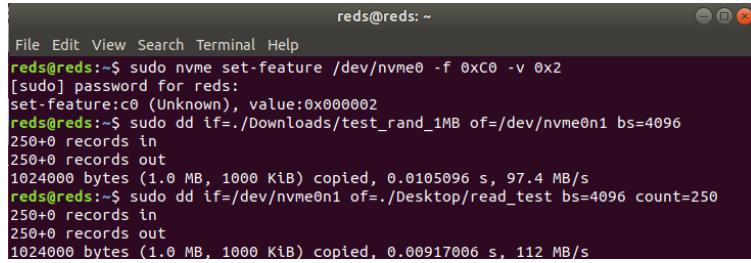


Figure 1.59: General Execution - AES-256 Encryption Waveform

Performance Tests

As in the previous cases, performance tests are carried out to evaluate IOPS, bandwidth and latency for the final computational storage configured to perform the cryptography.

As it can be seen in Figures 1.60 to 1.63, there is almost no degradation in performance after the insertion of the hardware accelerator.



```
red@reds:~$ sudo nvme set-feature /dev/nvme0 -f 0xC0 -v 0x2
[sudo] password for reds:
set feature:c0 (Unknown), value:0x000002
red@reds:~$ sudo dd if=~/Downloads/test_rand_1MB of=/dev/nvme0n1 bs=4096
250+0 records in
250+0 records out
1024000 bytes (1.0 MB, 1000 KiB) copied, 0.0105096 s, 97.4 MB/s
red@reds:~$ sudo dd if=/dev/nvme0n1 of=~/Desktop/read_test bs=4096 count=250
250+0 records in
250+0 records out
1024000 bytes (1.0 MB, 1000 KiB) copied, 0.00917006 s, 112 MB/s
```

Figure 1.60: Performance test - dd function, write and read - 128bit AXI DMA Version

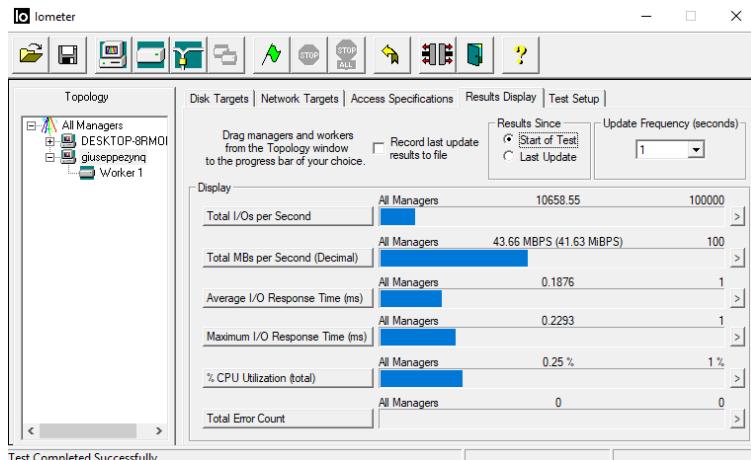


Figure 1.61: Performance test - Iometer Sequential Read transfer size = 4k, block size = 4k - Computational Storage

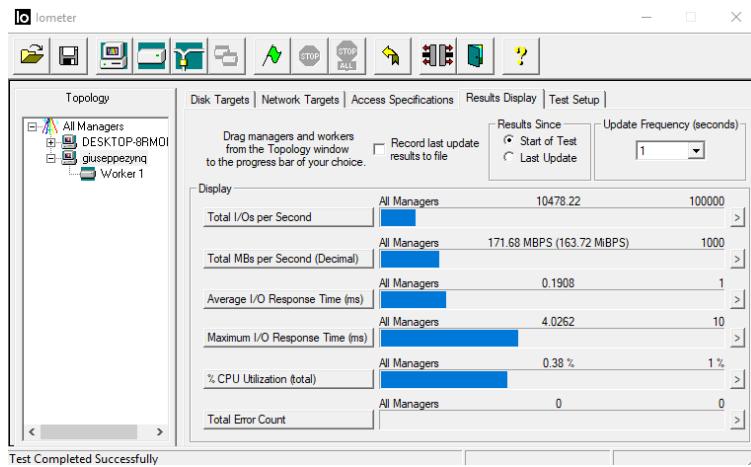


Figure 1.62: Performance test - Iometer Sequential Read transfer size = 16k, block size = 4k - Computational Storage

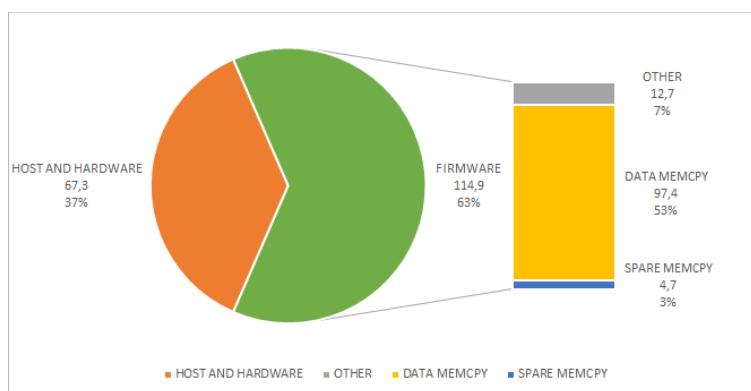


Figure 1.63: Performance test - Read Latency - Computational Storage

Bibliography

- [1] Yong Ho Song, *Cosmos+ OpenSSD 2017 Tutorial*, HYU ENC Lab of Hanyang University, 2017.
- [2] Xilinx, *ZC706 Evaluation Board for the Zynq-7000 XC7Z045 SoC, User Guide*, Page 46-47, 2019.
- [3] NVM Express, *NVM Express™ Base Specification Revision 1.4*, Page 206-207, June 2019.
- [4] Homer Hsing, *AES Core Specification*, OpenCores, February 2013, Accessed June 2020, <https://opencores.org/projects/tiny_aes>
- [5] Morris Dworkin, *Recommendation for Block Cipher Modes of Operation: Methods and Techniques*, NIST Special Publication 800-38A, December 2001.