Data Sheet
Jun. 19 – Ver. 1.0

POLARFIRE
NVMe Host Recorder IP

Features

The LDS NVME HOST RECORDER IP has been done for beginners and expert in NVMe to drive NVMe PCIe SSD.

The register file interface simplify the management of the IP for CPU interface or State Machine interface using APB bus:

- PCIe RP and EP register configuration is done automatically.
- NVMe register configuration is done automatically.
- Able to manage 8 Name Spaces.
- Able to manage until 16 IO Queue to fit specific user requirement.
  - Each IO Queue is independent.
- Able to manage 512Bytes and 4096Bytes sector size.
- Able to run nearly all Admin command in parallel of IO Queue.
- Many IO command already pre-defined to ease use of the IP.
- Configurable IO Queue buffer size to fit user memory requirement in case of small density FPGA: 256KB, 128KB, 64KB or 32KB.
- Able to read all PCIe RP and EP registers.
- Easy connection to Root Port PCIe IP through AXI bus.

When using a PCIe RP IP configured in Gen2 the system frequency is at 150MHz.
When using a PCIe RP IP configured in Gen3 the system frequency is at 250MHz.

The source code format is available for ease of customization. The customization can be done by Logic Design Solutions and **DO254** documentation is available on request.
This IP can be customized according to specific needs (application-specific requirement). Any other pre-designed functions can be integrated into the FPGA. FPGA density and I/O requirements can be defined according to customer specification.

Design Package

Device Family	**MICROCHIP PolarFire FPGA** **speed grade : 1**
Package file	**Source code or Source Encrypted.** Data Sheet, IP Interface Description and Constraint File.
Design Tool Used	MICROCHIP LIBERO SoC V12.0.
Support	Support provided by Logic Design Solutions 1 year e-mail and telephone support from Logic Design Solutions included in the IP price. Support does not cover User IP modifications. Maintenance Contract available.

Verification

The LDS NVME HOST RECORDER IP has been validated on the PolarFire Evaluation Kits (MPF300-EVAL_KIT) and several disks. List of disk available on request.

Performance

The demo provided makes a Disk Write and Read performance test on each disk connected.
A counter value is written on the disk and then read back and checked.
The performance depends on disk tested and memory configuration of the IP, especially for read performance.
Performance are better with higher memory configuration.
As an example:

- Clock frequency 150MHz.
- PCIe Gen2 x 4 HHHL disk.
- 100 Giga Byte data transfer.
- Sequential Write : 1060 MBytes/s
- Sequential Read : 1055 MBytes/s

Utilization

IP core only					
IP Configuration	**4LUT**	**USRAM 1K**	**FF**	**LSRAM 18K**	**Logic Element**
1 Queue – 32KBytes Buffer	12000	317	7550	20	12500
1 Queue – 64KBytes Buffer	12500	317	8020	33	13000
1 Queue – 128KBytes Buffer	13900	317	9400	71	14370
1 Queue – 256KBytes Buffer	16220	317	11700	135	16750
2 Queues – 32KBytes Buffer	15274	367	9900	33	15960
2 Queues – 64KBytes Buffer	16140	367	10840	59	16820
2 Queues – 128KBytes Buffer	19070	367	13600	135	19752
2 Queues – 256KBytes Buffer	23630	367	18220	263	24361

Until 16 Queues available.

TABLE OF CONTENTS

1. GENERAL DESCRIPTION 4

1.1 STATE MACHINE DEMO 4
1.2 CPU DEMO 6
1.3 POLARFIRE ROOT PORT PCIE IP 8
1.3.1 Block Diagram 8
1.3.2 Configuration of the PCIe IP in Libero Soc 9
1.4 LDS_STM_TEST MODULE 11
1.5 LDS_GEN_CHECK MODULE 12
1.5.1 Register map 12
1.6 TOP_NVME_HOST 14
1.7 TOP NVME HOST REGISTER MAP 15
1.7.1 Offset 00h: Configuration 16
1.7.2 Offset 04h: Control 17
1.7.3 Offset 08h: Status Admin 18
1.7.4 Offset 0Ch: Status IO 1 19
1.7.5 Offset 10h: Start Logical Block Low 20
1.7.6 Offset 14h: Start Logical Block High 20
1.7.7 Offset 18h: Number Logical Block Low 20
1.7.8 Offset 1Ch: Number Logical Block High 20
1.7.9 Offset 20h – 5Ch: NVMe Admin Command 16x32-Bits FIFO 20
1.7.10 Offset 60h – 6Ch: NVMe Admin Response 4x32-Bits FIFO 21
1.7.11 Offset 70h: Remaining Number of data to Transfer 21
1.7.12 Offset 74h: Status IO 2 22
1.7.13 Offset 78h: NVMe Admin Data Received FIFO – 4KBytes 22
1.7.14 Offset 7Ch: NVMe Admin Data Received Control 23
1.7.15 Offset 80h: NVMe Admin Data Transmit FIFO – 4KBytes 23
1.7.16 Offset 84h: NVMe Admin Data Transmit Control 23
1.7.17 Offset 88h: PCIe RP or PCIe EP address 23
1.7.18 Offset 8Ch: PCIe RP or PCIe EP read register 23

2. TOOL VERSION USED 24

3. RECOMMENDED DESIGN EXPERIENCE 24

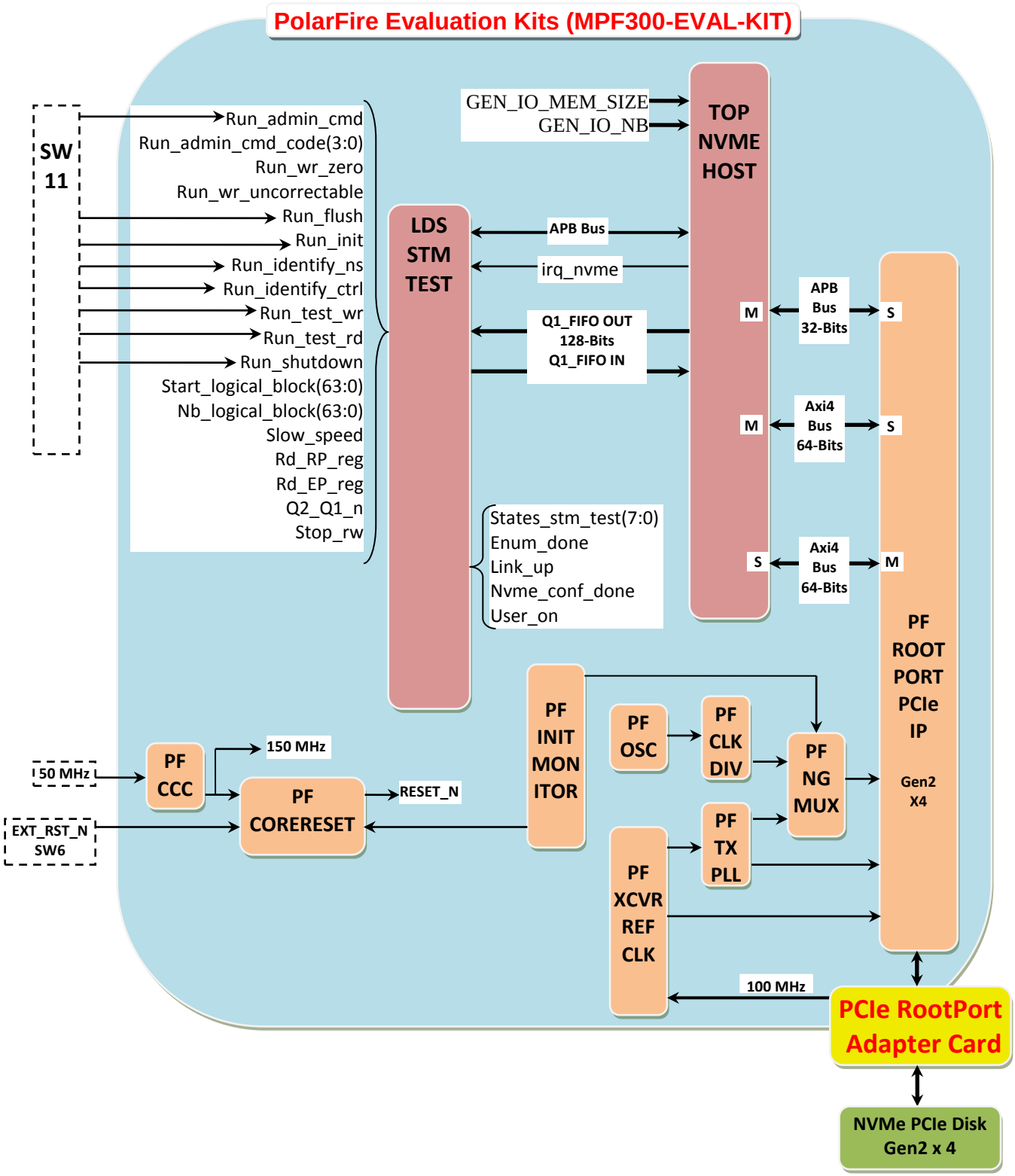
4. AVAILABLE SUPPORT PRODUCTS 24

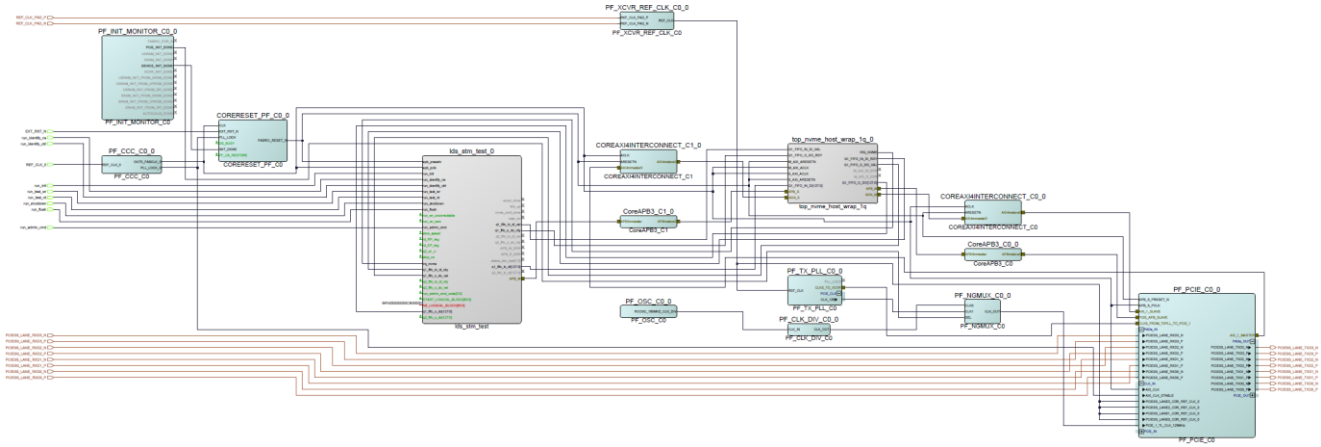
5. ORDERING INFORMATION 24

6. RELATED INFORMATION 24

1. General Description

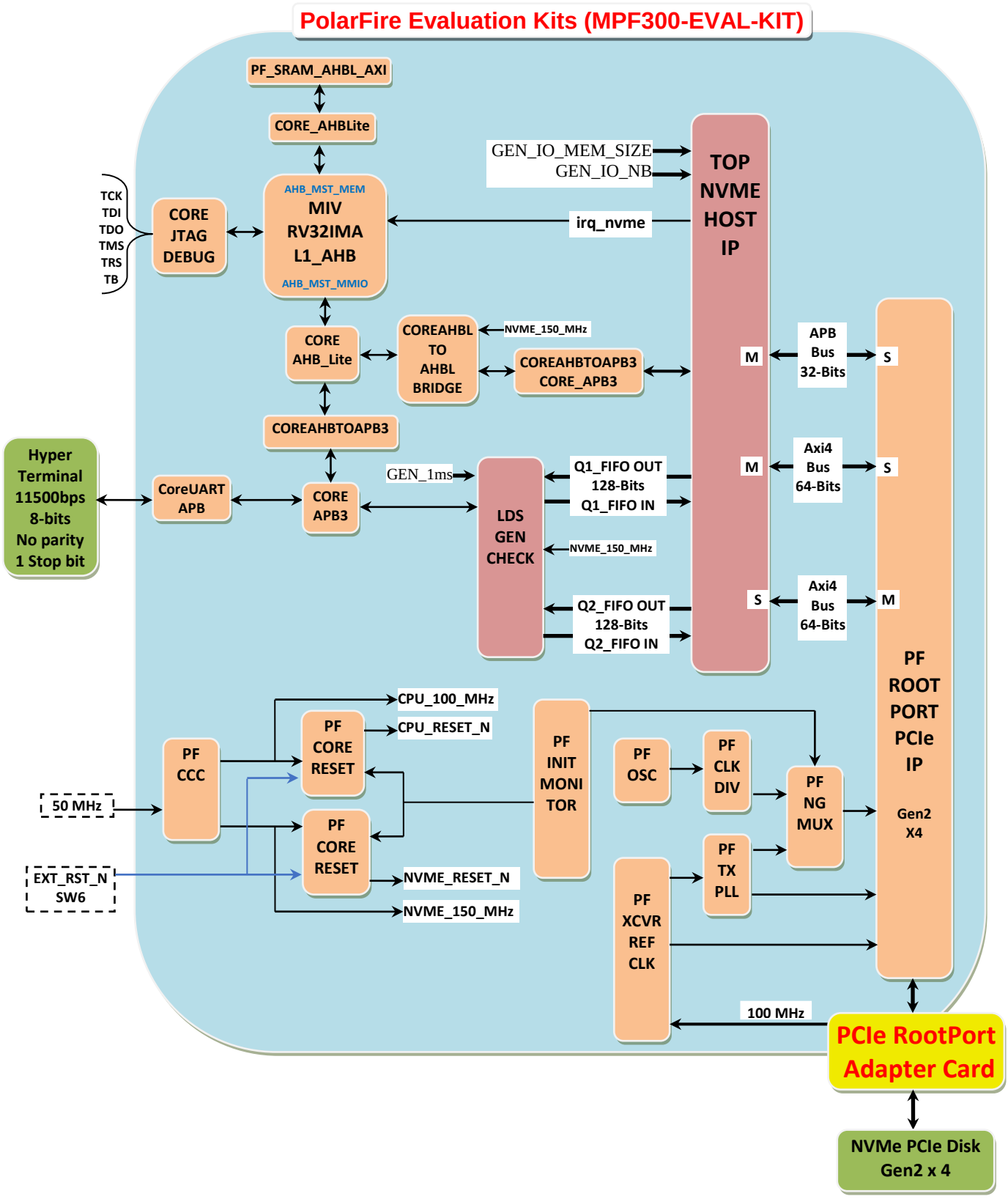
1.1 State Machine Demo

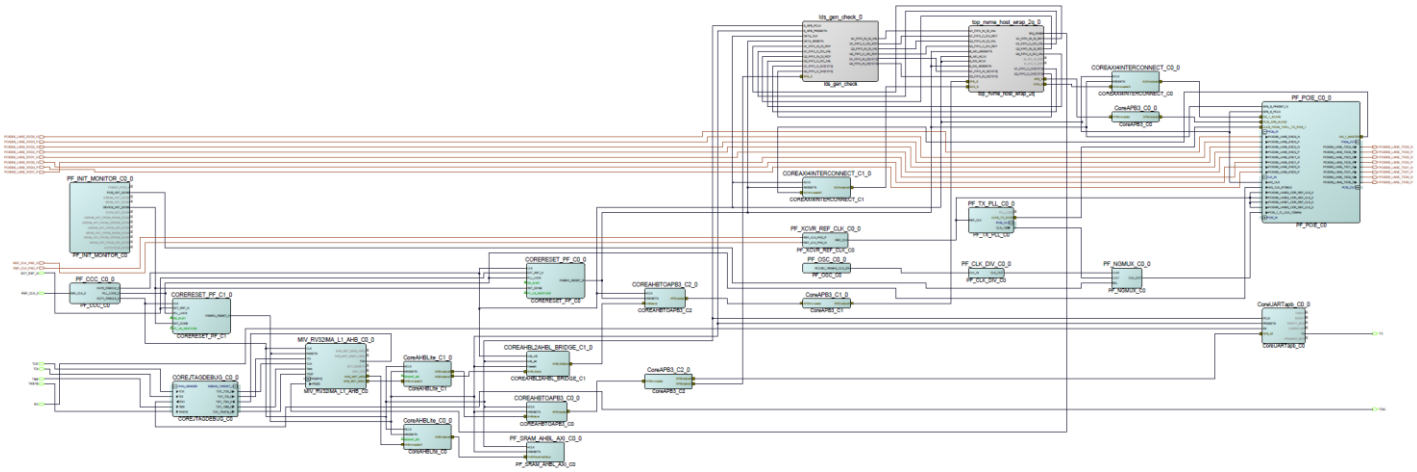




- The STM demo runs at 150MHz and its goal is to check performance during long transfer.
- You can run several commands through an external switch which run a state machine, please have a look on the demo documentation for details.
- The VHDL source code of the state machine is provided which enables the user to modify it to fit its project requirement.

1.2 CPU Demo

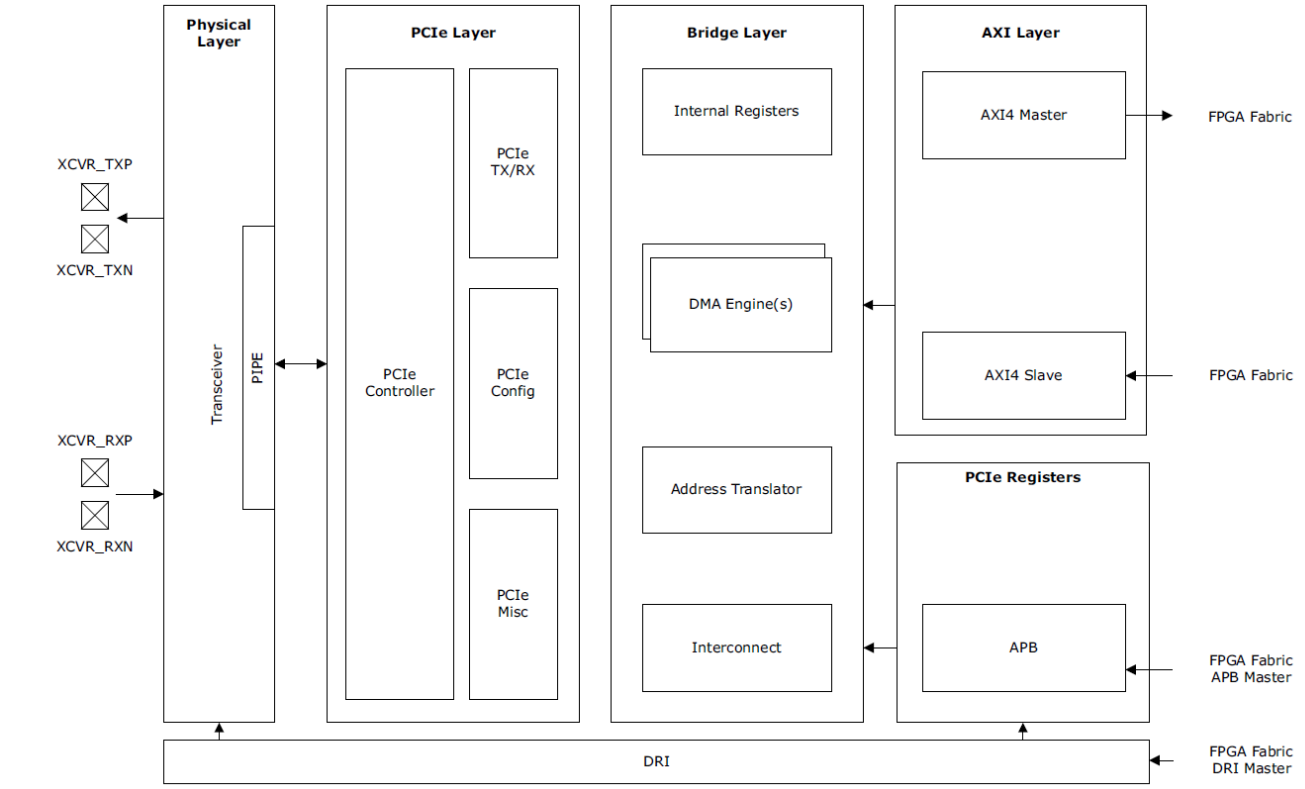

PolarFire Evaluation Kits (MPF300-EVAL-KIT)
PF_SRAM_AHBL_AXI
CORE_AHBLite
AHB_MST_MEM
MIV
RV32IMA
L1_AHB
AHB_MST_MMIO
TCK
TDI
TDO
TMS
TRS
TB
CORE
JTAG
DEBUG
GEN_IO_MEM_SIZE
GEN_IO_NB
irq_nvme
TOP
NVME
HOST
IP
CORE
AHB_Lite
COREAHBL
TO
AHBL
BRIDGE
NVME_150_MHz
COREAHBTOAPB3
CORE_APB3
M
APB
Bus
32-Bits
S
COREAHBTOAPB3
Hyper
Terminal
11500bps
8-bits
No parity
1 Stop bit
CoreUART
APB
CORE
APB3
GEN_1ms
LDS
GEN
CHECK
Q1_FIFO OUT
128-Bits
Q1_FIFO IN
NVME_150_MHz
Axi4
Bus
64-Bits
Q2_FIFO OUT
128-Bits
Q2_FIFO IN
PF
ROOT
PORT
PCIe
IP
Gen2
X4
CPU_100_MHz
CPU_RESET_N
PF
CCC
PF
CORE
RESET
PF
INIT
MONI
TOR
PF
OSC
PF
CLK
DIV
PF
NG
MUX
50 MHz
EXT_RST_N
SW6
PF
CORE
RESET
NVME_RESET_N
NVME_150_MHz
PF
TX
PLL
PF
XCVR
REF
CLK
100 MHz
PCIe RootPort
Adapter Card
NVMe PCIe Disk
Gen2 x 4



- The RISC-V CPU runs at 100MHz and the NVMe data runs at 150MHz. The goal of the demo is to check performance during long transfer.
- You can run a write and read performance test through a User Interface on the Hyper Terminal, please have a look on the demo documentation for details.
- The C source code is provided which enables the user to modify it to fit its project requirement.

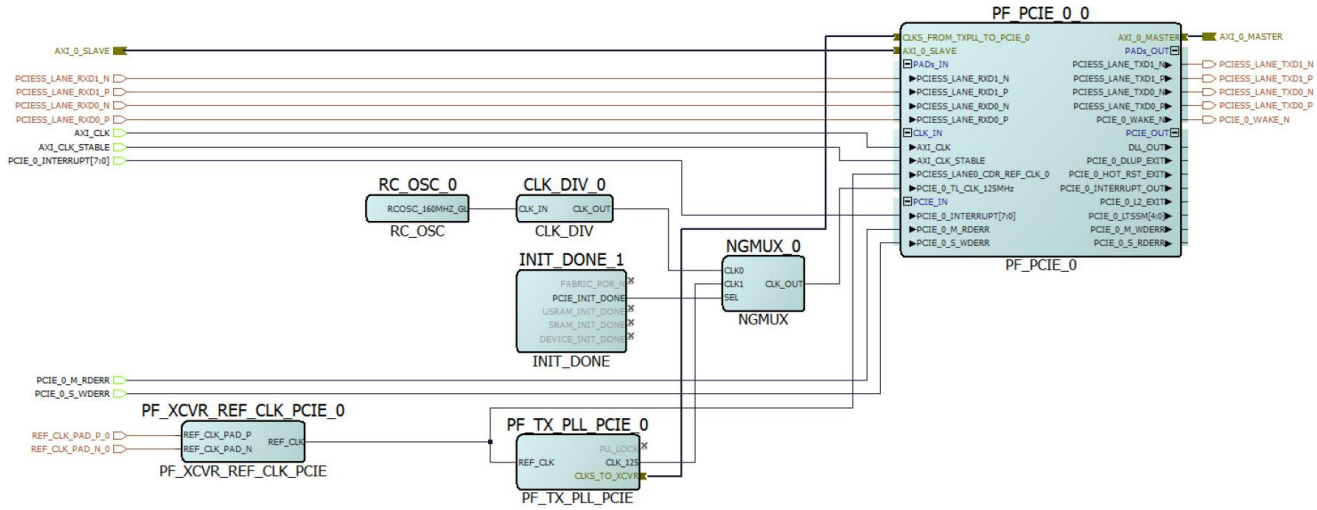
1.3 POLARFIRE ROOT PORT PCIe IP

1.3.1 Block Diagram



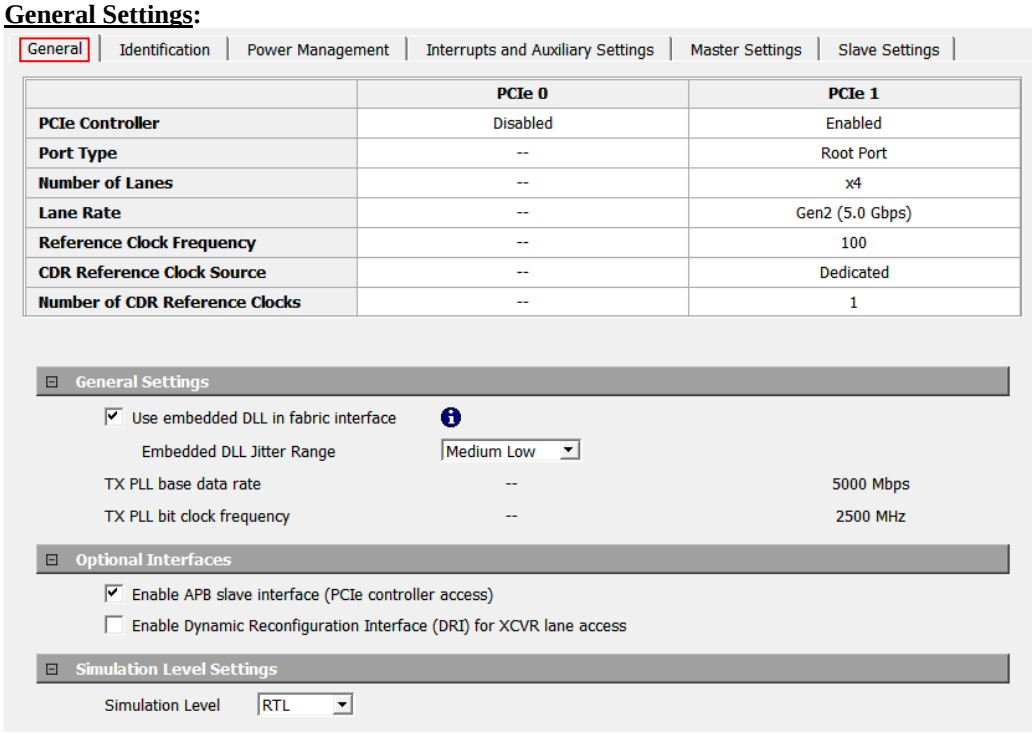
Please read **UG0685** to get details about this IP.

The SmartDesign representation of a complete PCIe example is the following:



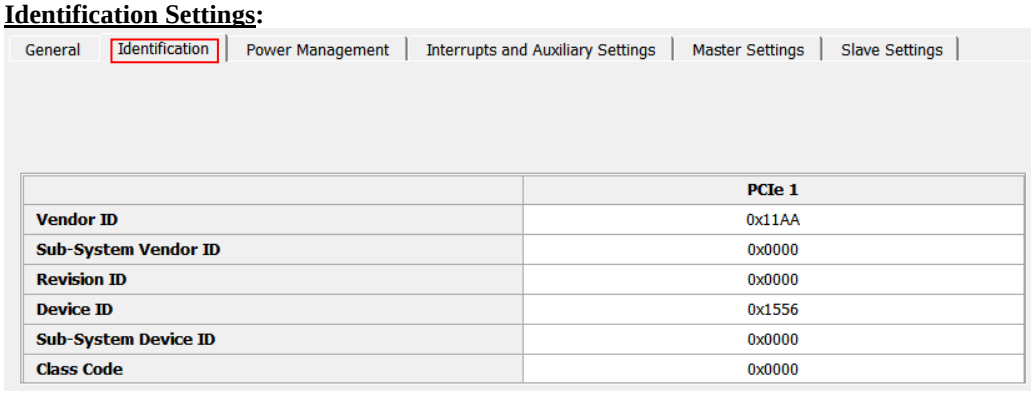
1.3.2 Configuration of the PCIe IP in Libero Soc

General Settings:



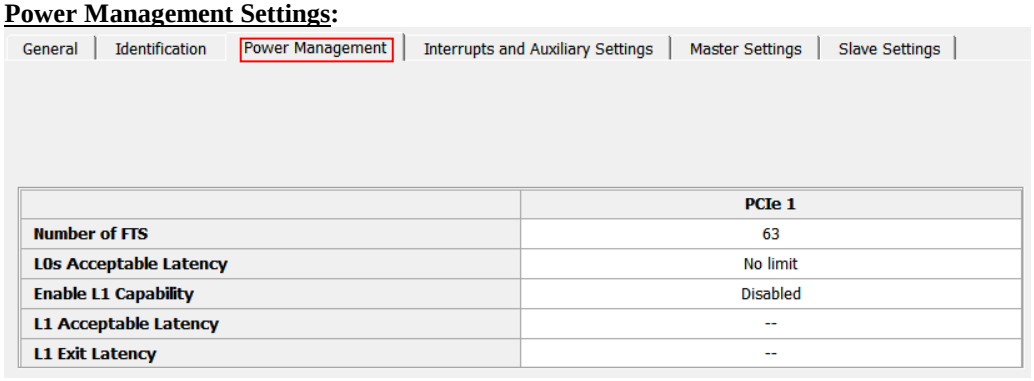
	PCIe 0	PCIe 1
PCIe Controller	Disabled	Enabled
Port Type	--	Root Port
Number of Lanes	--	x4
Lane Rate	--	Gen2 (5.0 Gbps)
Reference Clock Frequency	--	100
CDR Reference Clock Source	--	Dedicated
Number of CDR Reference Clocks	--	1

Identification Settings:



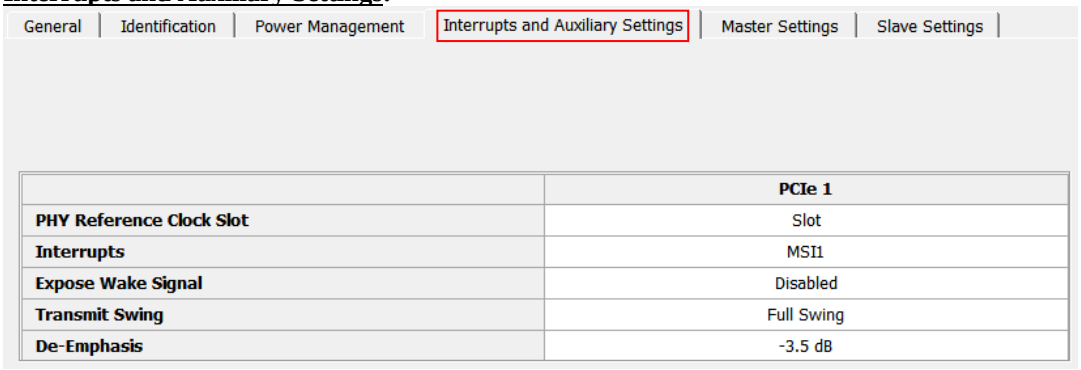
	PCIe 1
Vendor ID	0x11AA
Sub-System Vendor ID	0x0000
Revision ID	0x0000
Device ID	0x1556
Sub-System Device ID	0x0000
Class Code	0x0000

Power Management Settings:



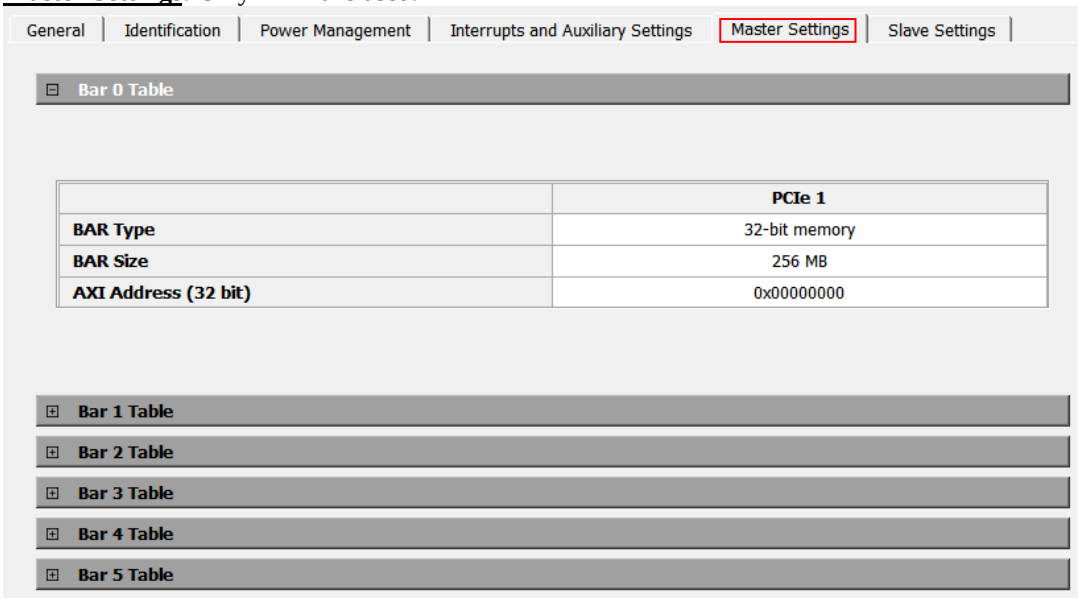
	PCIe 1
Number of FTS	63
L0s Acceptable Latency	No limit
Enable L1 Capability	Disabled
L1 Acceptable Latency	--
L1 Exit Latency	--

Interrupts and Auxiliary Settings:



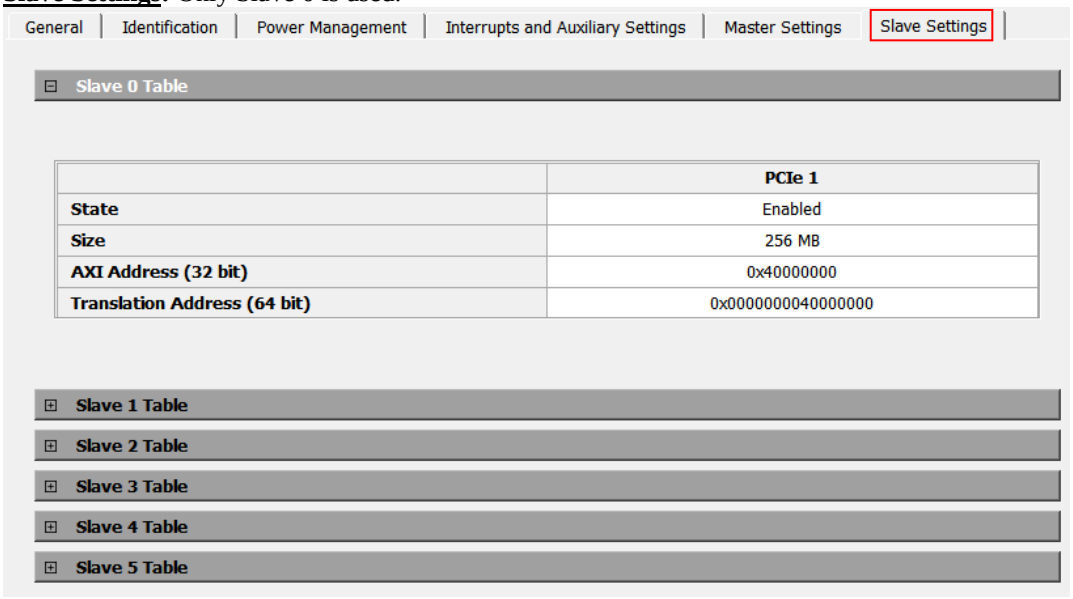
	PCIe 1
PHY Reference Clock Slot	Slot
Interrupts	MSI1
Expose Wake Signal	Disabled
Transmit Swing	Full Swing
De-Emphasis	-3.5 dB

Master Settings: Only BAR 0 is used.



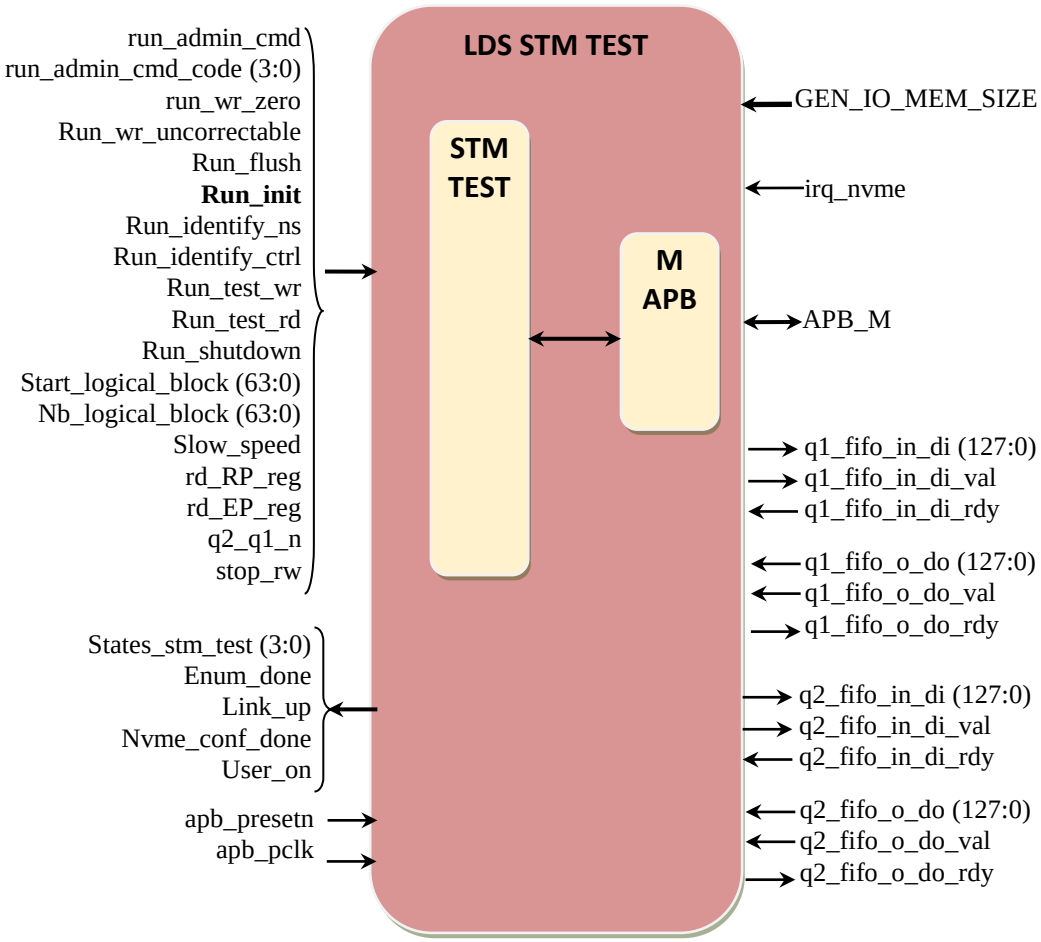
	PCIe 1
BAR Type	32-bit memory
BAR Size	256 MB
AXI Address (32 bit)	0x00000000

Slave Settings: Only Slave 0 is used.



	PCIe 1
State	Enabled
Size	256 MB
AXI Address (32 bit)	0x40000000
Translation Address (64 bit)	0x0000000040000000

1.4 LDS_STM_TEST Module



Keep Alive command	=> run_admin_cmd_code (3:0) = 0
GET_FEATURE_POWER_MANAGEMENT	=> run_admin_cmd_code (3:0) = 1
GET_LOG_PAGE_SMART_HEALTH_INFO	=> run_admin_cmd_code (3:0) = 2

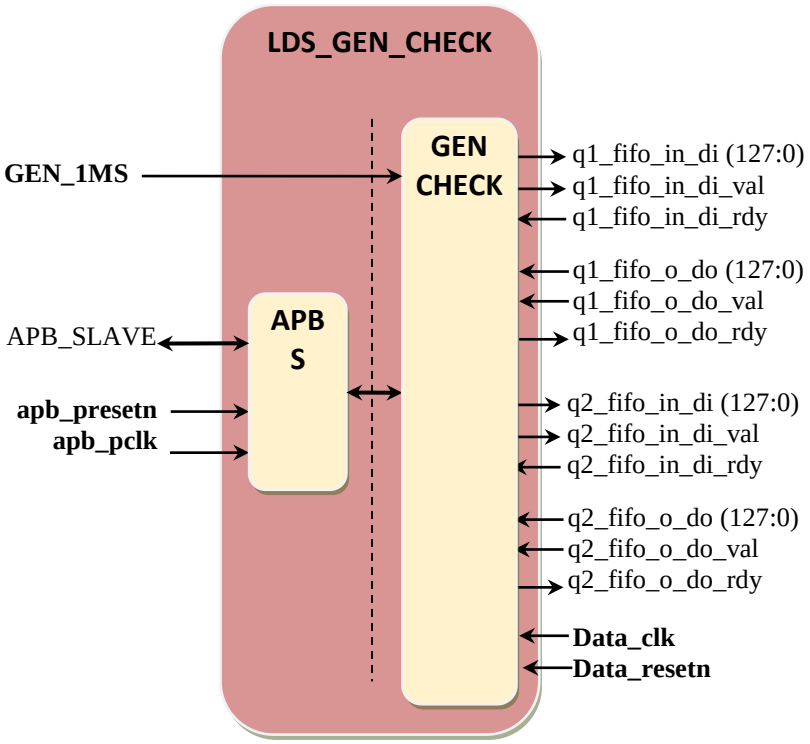
Other commands can be defined by the user.

A rising edge on one of the possible command runs it.
After power-up or disk shutdown the first command to run is **Run_init**.

The details of the State machine is provided in the demo data sheet.

The source code of this module is always provided. Hence, the user can use it for its project and modify it.

1.5 LDS_GEN_CHECK Module



The goal of this module is to generate a counter when writing to the disk and checking the value received when reading the disk.

When the "_rdy" signal goes low, the "_val" signal must go low as well immediately in a combinatorial way.

The registers are in the APB bus clock domain and the data test are in NVME_HOST_RECORDER IP clock domain.

This module is able to manage until two Queues.
The VHDL source Code of this module is always provided.

The definition of the Generics is the following:

GEN_1MS	integer := 150015; -- How many clock period to get 1ms at the frequency period - 150Mhz.

1.5.1 Register map

All register are 32-bits wide

Offset	Description
00h	Control Register
04h	Low Performance Counter Register
08h	High Performance Counter Register
0Ch	Performance Counter Register

1.5.1.1 Offset 00h: Control Register

Bits	Type	Reset	Description
31:7	RO	0	
06	RO	0	**Error on data received on Queue 2**: **0**: No error has been detected on received data. **1**: An error has been detected on received data. **Note**: This bit is set to 0 on the rising edge of bit 04 or bit 05.
05	R/W	0	**Generate Checking data on Queue 2**: **0**: Reset Read counter. **1**: Run Read counter.
04	R/W	0	**Generate write data on Queue 2**: **0**: Reset Write counter. **1**: Run Write counter.
03	R/W	0	**Generate Slow speed Queue 1**: **0**: Slow speed mode inactive. **1**: Slow speed mode active. **Note**: This mode generates data at 1/8 rate.
02	RO	0	**Error on data received on Queue 1**: **0**: No error has been detected on received data. **1**: An error has been detected on received data. **Note**: This bit is set to 0 on the rising edge of bit 0 or bit 1.
01	R/W	0	**Generate Checking data on Queue 1**: **0**: Reset Read counter. **1**: Run Read counter.
00	R/W	0	**Generate write data on Queue 1**: **0**: Reset Write counter. **1**: Run Write counter.

1.5.1.2 Offset 04h: Low Performance Counter Register

Bits	Type	Reset	Description
31:0	RO	0	**Low Value of a 64-Bits Counter running at the frequency of the bloc.** It enables to calculate the writing and the reading performance according the frequency of the bloc. In Gen2 the frequency is at 125MHz. In Gen3 the frequency is at 250MHz. **Note**: This register is set to 0 on the rising edge of bit 00 or bit 01 of Control Register.

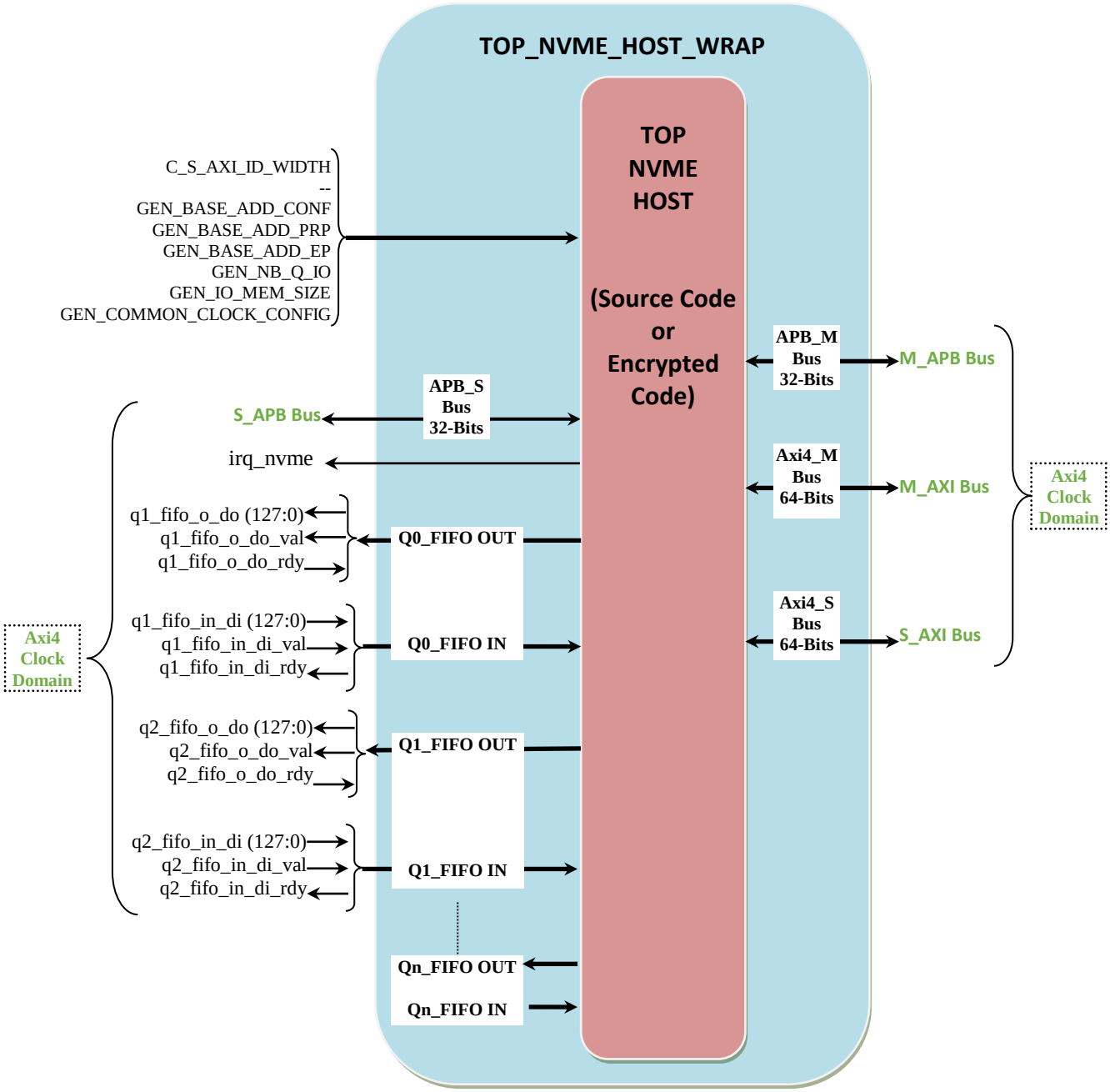
1.5.1.3 Offset 08h: High Performance Counter Register

Bits	Type	Reset	Description
31:0	RO	0	**High Value of a 64-Bits Counter running at the frequency of the bloc.** It enables to calculate the writing and the reading performance according the frequency of the bloc. In Gen2 the frequency is at 125MHz. In Gen3 the frequency is at 250MHz. **Note**: This register is set to 0 on the rising edge of bit 00 or bit 01 of Control Register.

1.5.1.4 Offset 0Ch: Performance Counter Register

Bits	Type	Reset	Description
31:0	RO	0	**Value of a 32-Bits Counter running at 1Khz => 1ms.** It enables to calculate the writing and the reading performance according the frequency of the bloc. **Note**: This register is set to 0 on the rising edge of bit 00 or bit 01 of Control Register.

1.6 TOP_NVME_HOST



Until 16 Queues can be defined by Generics GEN_NB_Q_IO.

When the "_rdy" signal goes low, the "_val" signal must go low as well immediately in a combinatorial way.

The S_APB Bus enable CPU or State Machine to access to the registers of the NVME-HOST-RECORDER IP.
All signals are on the same AXI4 clock domain.

The definition of the Generics is the following:

C_S_AXI_ID_WIDTH	integer := 5; -- Width of ID of AXI4 Slave Bus
GEN_BASE_ADD_CONF	integer := 0; -- PCIe Root IP Registers Base address in the CPU space.
GEN_BASE_ADD_PRP	integer := 1342177280; -- x"50000000" Internal Buffer PCIe Base Address
GEN_BASE_ADD_EP	integer := 1073741824 ; -- x"40000000" End Point PCIe Base Address
GEN_NB_Q_IO	integer := 1; -- Two Queues
GEN_IO_MEM_SIZE	integer := 32 -- 256KB / 128KB / 64KB / 32KB / Internal Buffer size per Queue
GEN_COMMON_CLOCK_CONFIG	integer := 1 – 1: Common PCIe Reference clock between Root Port and End Point

1.7 TOP NVME HOST Register map

All register are 32-bits wide

Offset	Description
00h	Configuration
04h	Control
08h	Status Admin
0Ch	Status IO 1
10h	Start Logical Block Low
14h	Start Logical Block High
18h	Number Logical Block Low
1Ch	Number Logical Block High
20h – 5Ch	NVMe Command – (16 x 32-Bits words) - Submission
60h-6Ch	NVMe Response – (4 x 32-Bits words) - Completion
70h	Remaining Number of data to Transfer
74h	Status IO 2
78h	NVMe Admin Data received FIFO – 4KBytes
7Ch	NVMe Admin Data received Control
80h	NVMe Admin Data to transmit FIFO– 4KBytes
84h	NVMe Admin Data to transmit Control
88h	PCIe RP or PCIe EP address register
8Ch	PCIe RP or PCIe EP read register

1.7.1 Offset 00h: Configuration

Bits	Type	Reset	Description
31:17	RO	0	
16	R/W	0	**Read Performance**: **0**: Max performance. The reading of data by the user must be at least at a rate superior than half of the maximum read throughput of the disk. **1**: Average performance. Select this option if the user will read data at a rate less than half of the maximum read throughput of the disk.
15:08	R/W	0	**LBA Data size supported:** The value is reported in terms of a power of two (2^n). A value smaller than 9 (i.e.) 512 Bytes is not supported. **00 to 08: not supported** **09:** 512bytes by default **LB_4K_512B_n=0** **0A:** 1KBytes **0B:** 2KBytes **0C:** 4KBytes. **LB_4K_512B_n=1** **0D:** 8KBytes. ... **Note**: This field is automatically updated at Power-up. This enable the user to know immediately the LBA data Size of the disk. Currently, the IP supports only 512 Bytes and 4KBytes selection.
07:04	R/W	0	**Number of IO Submission Queue and Completion Queue (NB_SQ_CQ_IO) requested:** Hardware dependent field. Minimum is 1 and maximum cannot be greater than **Number of Submission Queue and Completion Queue (NB_SQ_CQ_IO) available.** This is a 0's based value. (0=>1)
03:00	R	Impl. Spec	**Number of IO Submission Queue and Completion Queue (NB_SQ_CQ_IO) available:** Hardware dependent field. This is a 0's based value. (0=>1)

1.7.2 Offset 04h: Control

Bits	Type	Reset	Description
31:27	RO	0	Not used.
26	R/W		**STOP_RW:** Will stop the current read or write command executed in the **I/O QUEUE_NB.** This bit must be set to 1 only if the status bit **user_on** equal 1. This bit is set to 0 automatically.
25	R/W	0	**Reset_RW_Admin_Queue_Fifo**: When set to 1, will reset the Write and Read Admin Queue FIFO. This bit is set to 0 automatically.
24:22	R/W	0	**NSID**: Define the Name Space ID accessed. 000: NSID 1 001: NSID 2 .. 111: NSID 8 **Note**: If you need to manage more NS, please use the **RUN_ADMIN_CMD** option**.**
21	R/W	0	**RUN_IDENTIFY_NAMESPACE:** Will run an NVMe Identify command of the Namespace for the Admin Queue. The number of data received is 4KBytes. The user does not need to configure any other register for this command. This bit must be set to 1 only if the status bit **user_on** equal 1. This bit is set to 0 automatically.
20	R/W	0	**Reset_Read_Queue_Fifo**: When set to 1, will reset the Read Queue FIFO from the queue number indicated by the **Reset_Queue_Fifo_NB** register to be sure no remaining data from a previous reading are still in. This bit is set to 0 automatically.
19:16	R/W	0	**Reset_Queue_Fifo_NB**: 0: Queue 1 1: Queue 2 ... F: Queue 16
15	R/W	0	**Reset_Write_Queue_Fifo**: When set to 1, will reset the Write Queue FIFO from the queue number indicated by the **Reset_Queue_Fifo_NB** register to be sure no remaining data from a previous recording are still in. This bit is set to 0 automatically.
14	R/W	0	**RUN_WRITE_ZEROES:** Will run a write zero command for the **I/O QUEUE_NB,** starting from the **64-Bits Start Logical Block** register for a number of Logical Block defined in the **64-Bits Number Logical Block** register. This bit must be set to 1 only if the status bit **user_on** equal 1. This bit is set to 0 automatically. **Note**: A logical block may be deallocated with a Write Zeroes command.
13	R/W	0	**RUN_WRITE_UNCORRECTABLE:** Will run a write uncorrectable command for the **I/O QUEUE_NB,** starting from the **64-Bits Start Logical Block** register for only **16-Bits** number of Logical Block defined in the **Number Logical Block_low** register. This bit must be set to 1 only if the status bit **user_on** equal 1. This bit is set to 0 automatically.
12	R/W	0	**RUN_FLUSH:** Will run a Flush command for the **I/O QUEUE_NB.** This bit must be set to 1 only if the status bit **user_on** equal 1. This bit is set to 0 automatically.
11	R/W	0	**RUN_ADMIN_CMD:** Will run one Admin NVMe command defined in the **NVMe Admin Command** register. This bit must set to 1 only if the status bit **user_on** equal 1. This bit is set to 0 automatically. The number of data transfer required in the NVMe Command must never be bigger than 4KBytes.
10	R/W	0	**RUN_IDENTIFY_CONTROLLER:** Will run a NVMe Identify command of the Controller disk for the Admin Queue. The number of data received is 4KBytes. The user does not need to configure any other register for this command. This bit must be set to 1 only if the status bit **user_on** equal 1. This bit is set to 0 automatically.
09	R/W	0	**RUN_READ:** Will run a read command for the **I/O QUEUE_NB,** starting from the **64-Bits Start Logical Block** register for a number of Logical Block defined in the **64-Bits Number Logical Block** register. This bit must be set to 1 only if the status bit **user_on** equal 1. This bit is set to 0 automatically.
08	R/W	0	**RUN_WRITE:** Will run a write command for the **I/O QUEUE_NB,** starting from the **64-Bits Start Logical Block** register for a number of Logical Block defined in the **64-Bits Number Logical Block** register. This bit must be set to 1 only if the status bit **user_on** equal 1. This bit is set to 0 automatically.

07:04	R/W	0	**I/O QUEUE_NB**: indicates for which Submission/Completion Queue the command is related to. 0: I/O Queue 1 1: I/O Queue 2 ... F: I/O Queue 16
03:02	R	0	Not used.
01	R/W	0	**run_shutdown:** **0:** No effect. **1:** Run a disk Shut-down if the status bit **user_on** is set to 1. This bit is set to 0 automatically.
00	R/W	0	**run_user:** **1:** Enable the use of the NVMe block after a power up, a disk Shut-down or an error of PCIe enumeration. This bit is set to 0 automatically.

1.7.3 Offset 08h: Status Admin

Bits	Type	Reset	Description
31:10	RO	0	
09	RO	0	**admin_successful_command_completion**: **0:** **1:** Completion Status Field ok.
08	RO	0	**admin_fifo_cq_do_err**: **0:** **1:** One Completion admin queue error occurred.
07	RO	0	**disk_down**: **0:** Disk is not down. **1:** Disk is down. Wait for a **run_user** command.
06	RO	0	**nvme_conf_err**: **0:** No error during the NVMe Configuration. **1:** An error occurs during the NVMe Configuration.
05	RO	0	**Normal Shutdown_done** **0:** No Shutdown **1:** Normal shutdown reached
04	RWC 1	0	**admin_command_done:** **0:** Irq_nvme_0 is set to 0 **1:** Irq_nvme_0 is set to 1 A write with 1 on this bit will reset the bit.
03	RO	0	**user_on:** **0:** Command cannot be sent to the NVMe Block. **1:** Command can be sent to the NVMe Block.
02	RO	0	**nvme_conf_done:** **0:** NVMe Configuration not done. **1:** NVMe Configuration done
01	RO	0	**link_up:** **0:** PCIe not link up. **1:** PCIe link up.
00	RO	0	**enum_done:** **0:** PCIe Enumeration not done. **1:** PCIe Enumeration done

A read of the register clears the bits except for the RWC1 bit 04.

1.7.4 Offset 0Ch: Status IO 1

Bits	Type	Reset	Description
31	RWC1	0	**q8_fifo_sq_do_err**: One Submission IO queue error occurred.
30	RWC1	0	**q8_successful_command_completion**: Completion Status Field ok
29	RWC1	0	**q8_fifo_cq_do_err**: One Completion IO queue error occurred.
28	RWC1	0	**q8_command_done:** A write with 1 on this bit will reset the bit. **0:** Irq nvme_1 is set to 0 **1:** Irq nvme_1 is set to 1
27	RWC1	0	**q7_fifo_sq_do_err**: One Submission IO queue error occurred.
26	RWC1	0	**q7_successful_command_completion**: Completion Status Field ok
25	RWC1	0	**q7_fifo_cq_do_err**: One Completion IO queue error occurred.
24	RWC1	0	**q7_command_done:** A write with 1 on this bit will reset the bit. **0:** Irq nvme_1 is set to 0 **1:** Irq nvme_1 is set to 1
23	RWC1	0	**q6_fifo_sq_do_err**: One Submission IO queue error occurred.
22	RWC1	0	**q6_successful_command_completion**: Completion Status Field ok
21	RWC1	0	**q6_fifo_cq_do_err**: One Completion IO queue error occurred.
20	RWC1	0	**q6_command_done:** A write with 1 on this bit will reset the bit. **0:** Irq nvme_1 is set to 0 **1:** Irq nvme_1 is set to 1
19	RWC1	0	**q5_fifo_sq_do_err**: One Submission IO queue error occurred.
18	RWC1	0	**q5_successful_command_completion**: Completion Status Field ok
17	RWC1	0	**q5_fifo_cq_do_err**: One Completion IO queue error occurred.
16	RWC1	0	**q5_command_done:** A write with 1 on this bit will reset the bit. **0:** Irq nvme_1 is set to 0 **1:** Irq nvme_1 is set to 1
15	RWC1	0	**q4_fifo_sq_do_err**: One Submission IO queue error occurred.
14	RWC1	0	**q4_successful_command_completion**: Completion Status Field ok
13	RWC1	0	**q4_fifo_cq_do_err**: One Completion IO queue error occurred.
12	RWC1	0	**q4_command_done:** A write with 1 on this bit will reset the bit. **0:** Irq nvme_1 is set to 0 **1:** Irq nvme_1 is set to 1
11	RWC1	0	**q3_fifo_sq_do_err**: One Submission IO queue error occurred.
10	RWC1	0	**q3_successful_command_completion**: Completion Status Field ok
09	RWC1	0	**q3_fifo_cq_do_err**: One Completion IO queue error occurred.
08	RWC1	0	**q3_command_done:** A write with 1 on this bit will reset the bit. **0:** Irq nvme_1 is set to 0 **1:** Irq nvme_1 is set to 1
07	RWC1	0	**q2_fifo_sq_do_err**: One Submission IO queue error occurred.
06	RWC1	0	**q2_successful_command_completion**: Completion Status Field ok
05	RWC1	0	**q2_fifo_cq_do_err**: One Completion IO queue error occurred.
04	RWC1	0	**q2_command_done:** A write with 1 on this bit will reset the bit. **0:** Irq nvme_1 is set to 0 **1:** Irq nvme_1 is set to 1
03	RWC1	0	**q1_fifo_sq_do_err**: One Submission IO queue error occurred.
02	RWC1	0	**q1_successful_command_completion**: Completion Status Field ok
01	RWC1	0	**q1_fifo_cq_do_err**: One Completion IO queue error occurred.
00	RWC1	0	**q1_command_done:** A write with 1 on this bit will reset the bit. **0:** Irq nvme_1 is set to 0 **1:** Irq nvme_1 is set to 1

1.7.5 Offset 10h: Start Logical Block Low

Bits	Type	Reset	Description
31:00	R/W	0	**Start Logical Block Low:**

1.7.6 Offset 14h: Start Logical Block High

Bits	Type	Reset	Description
31:00	R/W	0	**Start Logical Block High:**

1.7.7 Offset 18h: Number Logical Block Low

Bits	Type	Reset	Description
31:00	R/W	0	**Number Logical Block Low:**

1.7.8 Offset 1Ch: Number Logical Block High

Bits	Type	Reset	Description
31:00	R/W	0	**Number Logical Block High:**

1.7.9 Offset 20h – 5Ch: NVMe Admin Command 16x32-Bits FIFO

Bits	Type	Reset	Description
31:00	W	0	**Add 20h : NVMe Admin Command 32-Bits word 0**
31:00	W	0	**Add 24h: NVMe Admin Command 32-Bits word 1**
31:00	W	0	**Add 28h: NVMe Admin Command 32-Bits word 2**
31:00	W	0	**Add 2Ch: NVMe Admin Command 32-Bits word 3**
31:00	W	0	**Add 30h: NVMe Admin Command 32-Bits word 4**
31:00	W	0	**Add 34h: NVMe Admin Command 32-Bits word 5**
31:00	W	0	**Add 38h: NVMe Admin Command 32-Bits word 6**
31:00	W	0	**Add 3Ch: NVMe Admin Command 32-Bits word 7**
31:00	W	0	**Add 40h: NVMe Admin Command 32-Bits word 8**
31:00	W	0	**Add 44h: NVMe Admin Command 32-Bits word 9**
31:00	W	0	**Add 48h: NVMe Admin Command 32-Bits word 10**
31:00	W	0	**Add 4Ch: NVMe Admin Command 32-Bits word 11**
31:00	W	0	**Add 50h: NVMe Admin Command 32-Bits word 12**
31:00	W	0	**Add 54h: NVMe Admin Command 32-Bits word 13**
31:00	W	0	**Add 58h : NVMe Admin Command 32-Bits word 14**
31:00	W	0	**Add 5Ch : NVMe Admin Command 32-Bits word 15**

To use this register, the user should know the NVMe Admin Command description from the NVM Express revision 1.3 specification available for download at http://nvmexpress.org
It enable the user to run any kind of admin command using this register and

- NVMe Admin response register
- NVMe Admin Data Received FIFO
- NVMe Admin Data Received Control
- NVMe Admin Data Transmit FIFO
- NVMe Admin Data Transmit Control

An example of use is described in the C source code provided with the CPU demo.

1.7.10 Offset 60h – 6Ch: NVMe Admin Response 4x32-Bits FIFO

Bits	Type	Reset	Description
31:00	R	0	**Add 60h: NVMe Admin Response 32-Bits word 0**
31:00	R	0	**Add 64h: NVMe Admin Response 32-Bits word 1**
31:00	R	0	**Add 68h: NVMe Admin Response 32-Bits word 2**
31:00	R	0	**Add 6Ch : NVMe Admin Response 32-Bits word 3**

1.7.11 Offset 70h: Remaining Number of data to Transfer

Bits	Type	Reset	
31:00	RO	0	Enable to follow the remaining number of data to transfer on Queue IO 1 only.

1.7.12 Offset 74h: Status IO 2

Bits	Type	Reset	Description
31	RWC1	0	**q16_fifo_sq_do_err**: One Submission IO queue error occurred.
30	RWC1	0	**q16_successful_command_completion**: Completion Status Field ok
29	RWC1	0	**q16_fifo_cq_do_err**: One Completion IO queue error occurred.
28	RWC1	0	**q16_command_done:** A write with 1 on this bit will reset the bit. **0:** Irq nvme_1 is set to 0 **1:** Irq nvme_1 is set to 1
27	RWC1	0	**q15_fifo_sq_do_err**: One Submission IO queue error occurred.
26	RWC1	0	**q15_successful_command_completion**: Completion Status Field ok
25	RWC1	0	**q15_fifo_cq_do_err**: One Completion IO queue error occurred.
24	RWC1	0	**q15_command_done:** A write with 1 on this bit will reset the bit. **0:** Irq nvme_1 is set to 0 **1:** Irq nvme_1 is set to 1
23	RWC1	0	**q14_fifo_sq_do_err**: One Submission IO queue error occurred.
22	RWC1	0	**q14_successful_command_completion**: Completion Status Field ok
21	RWC1	0	**q14_fifo_cq_do_err**: One Completion IO queue error occurred.
20	RWC1	0	**q14_command_done:** A write with 1 on this bit will reset the bit. **0:** Irq nvme_1 is set to 0 **1:** Irq nvme_1 is set to 1
19	RWC1	0	**q13_fifo_sq_do_err**: One Submission IO queue error occurred.
18	RWC1	0	**q13_successful_command_completion**: Completion Status Field ok
17	RWC1	0	**q13_fifo_cq_do_err**: One Completion IO queue error occurred.
16	RWC1	0	**q13_command_done:** A write with 1 on this bit will reset the bit. **0:** Irq nvme_1 is set to 0 **1:** Irq nvme_1 is set to 1
15	RWC1	0	**q12_fifo_sq_do_err**: One Submission IO queue error occurred.
14	RWC1	0	**q12_successful_command_completion**: Completion Status Field ok
13	RWC1	0	**q12_fifo_cq_do_err**: One Completion IO queue error occurred.
12	RWC1	0	**q12_command_done:** A write with 1 on this bit will reset the bit. **0:** Irq nvme_1 is set to 0 **1:** Irq nvme_1 is set to 1
11	RWC1	0	**q11_fifo_sq_do_err**: One Submission IO queue error occurred.
10	RWC1	0	**q11_successful_command_completion**: Completion Status Field ok
09	RWC1	0	**q11_fifo_cq_do_err**: One Completion IO queue error occurred.
08	RWC1	0	**q11_command_done:** A write with 1 on this bit will reset the bit. **0:** Irq nvme_1 is set to 0 **1:** Irq nvme_1 is set to 1
07	RWC1	0	**q10_fifo_sq_do_err**: One Submission IO queue error occurred.
06	RWC1	0	**q10_successful_command_completion**: Completion Status Field ok
05	RWC1	0	**q10_fifo_cq_do_err**: One Completion IO queue error occurred.
04	RWC1	0	**q10_command_done:** A write with 1 on this bit will reset the bit. **0:** Irq nvme_1 is set to 0 **1:** Irq nvme_1 is set to 1
03	RWC1	0	**q9_fifo_sq_do_err**: One Submission IO queue error occurred.
02	RWC1	0	**q9_successful_command_completion**: Completion Status Field ok
01	RWC1	0	**q9_fifo_cq_do_err**: One Completion IO queue error occurred.
00	RWC1	0	**q9_command_done:** A write with 1 on this bit will reset the bit. **0:** Irq nvme_1 is set to 0 **1:** Irq nvme_1 is set to 1

1.7.13 Offset 78h: NVMe Admin Data Received FIFO – 4KBytes

Bits	Type	Reset	Description
31:00	RO	X	Data received from an Admin NVMe Command. For example after an Identify command. The user should read the 4KB data completely.

1.7.14 Offset 7Ch: NVMe Admin Data Received Control

Bits	Type	Reset	Description
31:02	RO	0	Not used.
1	WO	0	**NVMe Data Received FIFO reset**: **1**: Reset the **NVMe Data Received FIFO** **0**: No action done **Note**: the user should write a '1' in this bit, each time he needs to reset the **NVMe Data Received FIFO.** The Bit is reset automatically**.**
0	RO	1	**NVMe Data Received FIFO empty**: **1**: Empty **0**: Not empty

1.7.15 Offset 80h: NVMe Admin Data Transmit FIFO – 4KBytes

Bits	Type	Reset	Description
31:00	WO	X	Data to transmit from any Admin NVMe Command which needs data to send to the disk.

1.7.16 Offset 84h: NVMe Admin Data Transmit Control

Bits	Type	Reset	Description
31:03	RO	0	Not used.
2	WO	0	**NVMe Data Transmit FIFO reset**: **1**: Reset the **NVMe Data Transmit FIFO** **0**: No action done **Note**: the user should write a '1' in this bit, each time he needs to reset the **NVMe Data Transmit FIFO.** The Bit is reset automatically**.**
1	RO	1	**NVMe Data Transmit FIFO full**: **1**: Full **0**: Not full
0	RO	1	**NVMe Data Transmit FIFO empty**: **1**: Empty **0**: Not empty

1.7.17 Offset 88h: PCIe RP or PCIe EP address

Bits	Type	Reset	Description
31:03	RO	0	Not used.
18	RO	0	**Read of a PCIe register done** **1:** Set to 1 when the PCIe read register has been updated. **0:** Set to 0 when the 'Request a read of a PCIe register' bit is set to 1.
17	R/W	0	**Request a read of a PCIe register:** **1:** Request a read of the PCIe register according Bit 16 and Bits(15:0) **0:** Set to 0 when the **Read of a PCIe register done**' bit is set to 1. **Note**: A request can be done only if the bit **enum_done** from the **Status Admin register** is set to 1.
16	R/W	0	**RP_EP:** **1:** End Point register set selected **0:** Root Port register set selected
15:0	R/W	0	**PCIe RP or PCIe EP address:** **Define the address offset of the register to read**

1.7.18 Offset 8Ch: PCIe RP or PCIe EP read register

Bits	Type	Reset	Description
31:00	RO	0	Value of the PCIe RP or EP register read. The value is valid when the ' **Read of a PCIe register done**' bit is set to 1.

2. Tool version used

The IP has been done with a PC on Windows 7 and with the following tools:

- LIBERO SoC V12.0

Every tool has been used through its GUI.

3. Recommended Design Experience

Designers should be familiar with PCIe, VHDL, synthesis tools, FPGA Place and Route data flow and VHDL simulation software. Experience with microprocessor is recommended. The IP can be easily integrated into hierarchical VHDL designs.

4. Available Support Products

Support products available from Logic Design Solutions.

5. Ordering Information

To purchase or make further inquiries about this, or any other Logic Design Solutions products and services, contact Logic Design Solutions in France.
Logic Design Solutions also offers macro integration and design services on FPGA technologies.
Logic Design Solutions macros are purchased under a License Agreement, copies of which are available on request.
Logic Design Solutions reserves the right to make changes to these specifications at any time, without notice.

6. Related Information

Logic Design Solutions
106 Boulevard de la Résistance – 93460 Gournay sur Marne – France.
Phone : +33 (0) 1 45 92 24 47
E-mail : info@logic-design-solutions.com
WEB: http://www.logic-design-solutions.com