HCTL-IP reference design manual

Rev1.5 6-Jul-23

HCTL-IP reference design manual

Rev1.5 6-Jul-23

1 Introduction

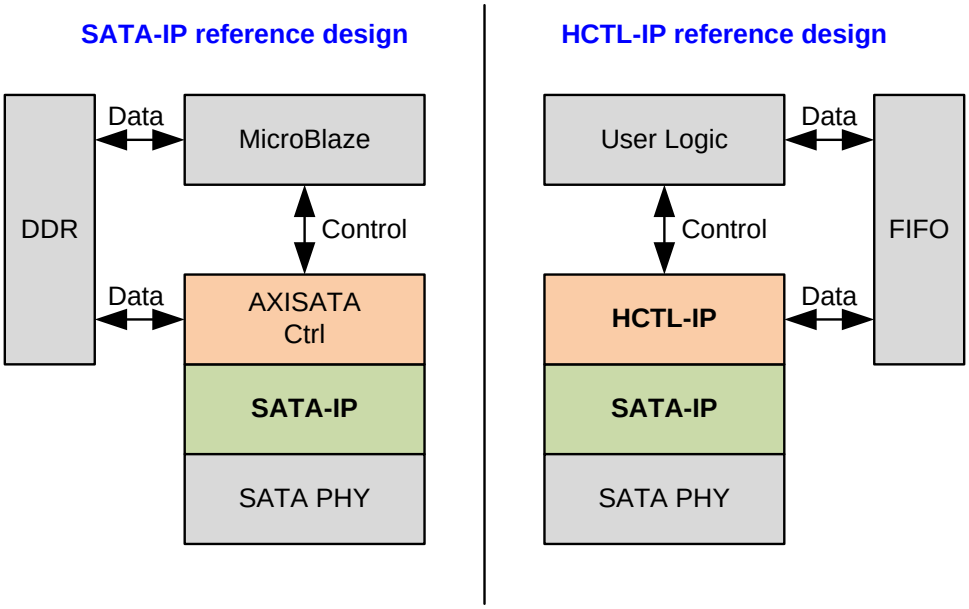

Figure 1-1 HCTL-IP hardware comparing to SATA-IP hardware

In SATA-IP reference design, the application layer is designed by using firmware running on MicroBlaze and SATA FIS packet is stored to DDR. By using HCTL-IP, the reference design is simpler. Application layer of SATA protocol is implemented by HCTL-IP, so CPU and DDR do not need in the design. User Logic to run with HCTL-IP can be designed by simple state machine. Data buffer uses only FIFO to store user data instead of DDR to store FIS packet.

The advantage of using MicroBlaze firmware on SATA-IP reference design is system flexibility. The firmware supports many SATA commands and the additional command can be supported by updating the firmware. But the command on HCTL-IP is fixed to four commands, i.e. Identify device, Security erase, Write, and Read.

The advantage of using HCTL-IP is less resource utilization and higher performance. DDR and MicroBlaze use much logic resource to implement while HCTL-IP design uses only FIFO to be data buffer. For performance issue, the overhead time to process one SATA packet with SATA-IP by CPU firmware is more than HCTL-IP. So, test result of Write command on HCTL-IP reference design shows better speed than SATA-IP reference design.

More details of SATA-IP reference design are described in following documents.
http://www.dgway.com/products/IP/SATA-IP/dg_sata_ip_refdesign_host_7series_en.pdf
http://www.dgway.com/products/IP/SATA-IP/dg_sata_ip_host_demo_instruction_7series_en.pdf

2 Hardware overview

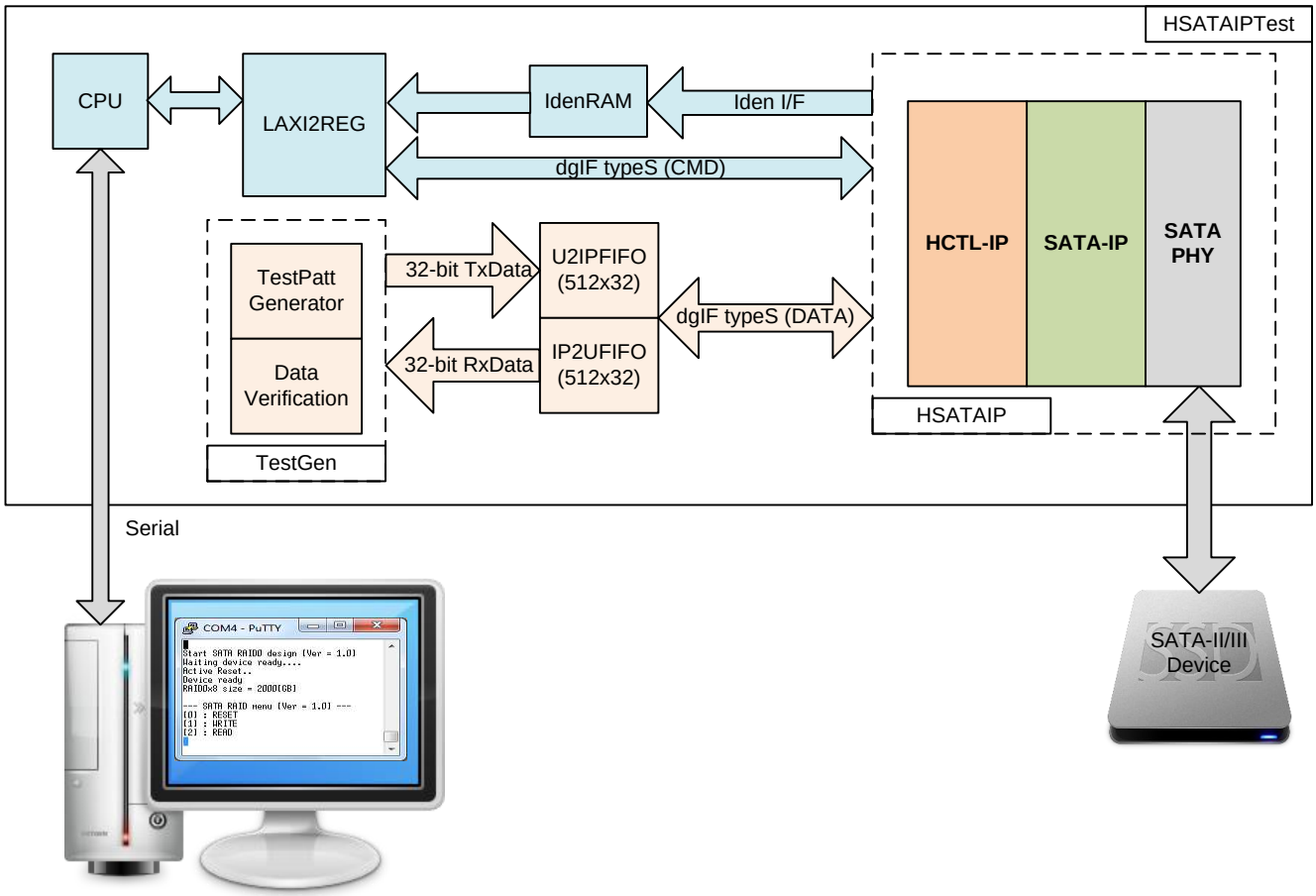

Figure 2-1 HCTL-IP demo hardware

The hardware system can be split into three groups following the interface i.e.

1) TestGen: The example of user logic to write and read data in this reference design is TestGen module. TestGen module generates test data to U2IPFIFO at the highest speed with flow control in Write command. For Read command, TestGen reads and verifies test data from IP2UFIFO at the highest speed with flow control. TestGen uses 32-bit data bus and runs in UserClk domain which is equal to 200 MHz. So, maximum bandwidth is 800 MB/s which is more than maximum performance of SATA-III devices.
2) SATA: HCTL-IP connecting with SATA-IP and SATA PHY is designed to interface with SATA device. Command and data interface of HCTL-IP is dgIF typeS format. For Identify device data, the interface is designed to store data to RAM.
3) CPU: Test operation in the demo is controlled by user through Serial console. CPU firmware is designed to receive test parameters and the command from user. CPU sets parameters to the hardware through AXI4-Lite bus. LAxi2Reg has the register sets of test parameters which are mapped to different address of CPU. LAxi2Reg decodes the address of AXI4-Lite bus to select the active parameter. For write access, Write data from AXI4-Lite bus is set to the selected parameter following the address. For read access, Read data from selected parameter is returned to AXI4-Lite bus. Read access is applied for CPU monitoring and displaying the hardware status to the user through Serial console.

   More details of the hardware are described as follows.

2.1 TestGen

This module is designed to generate Test pattern to WrFf in Write command or reads data from RdFf to verify in Read command at the fastest speed to check system performance. The details of hardware inside TestGen are shown in Figure 2-2.

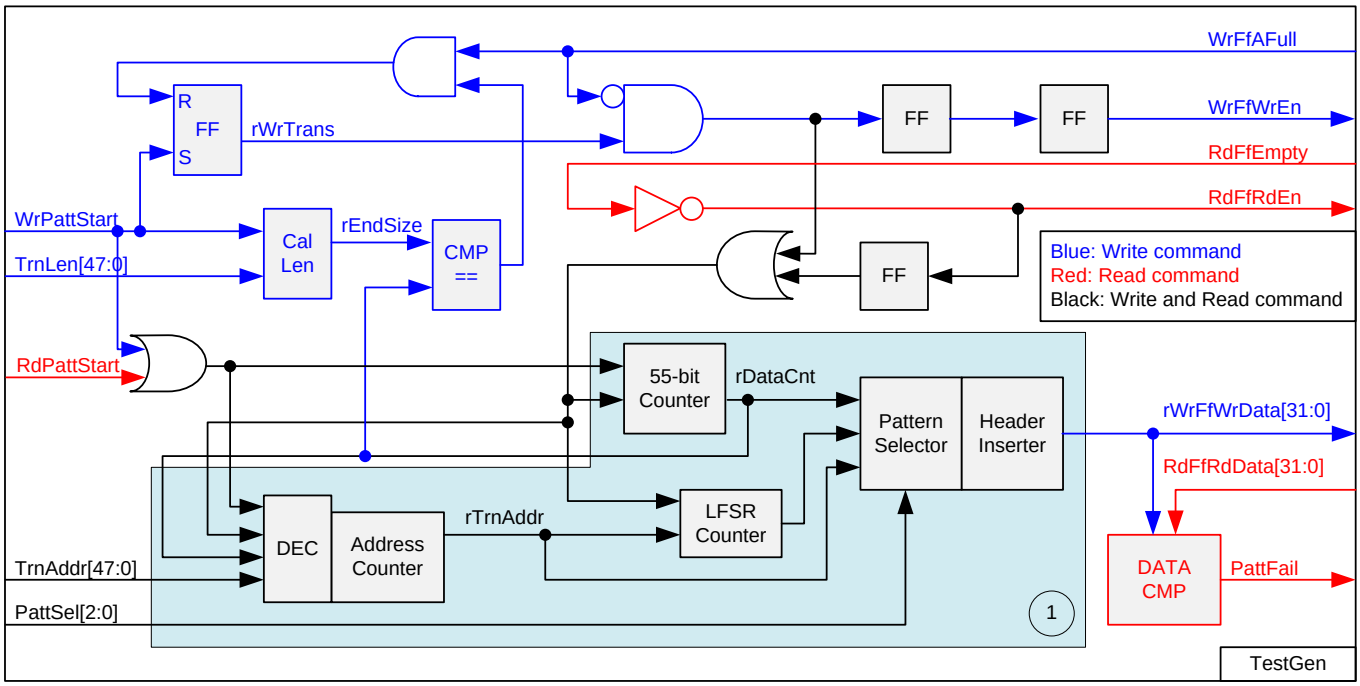

Figure 2-2 TestGen hardware

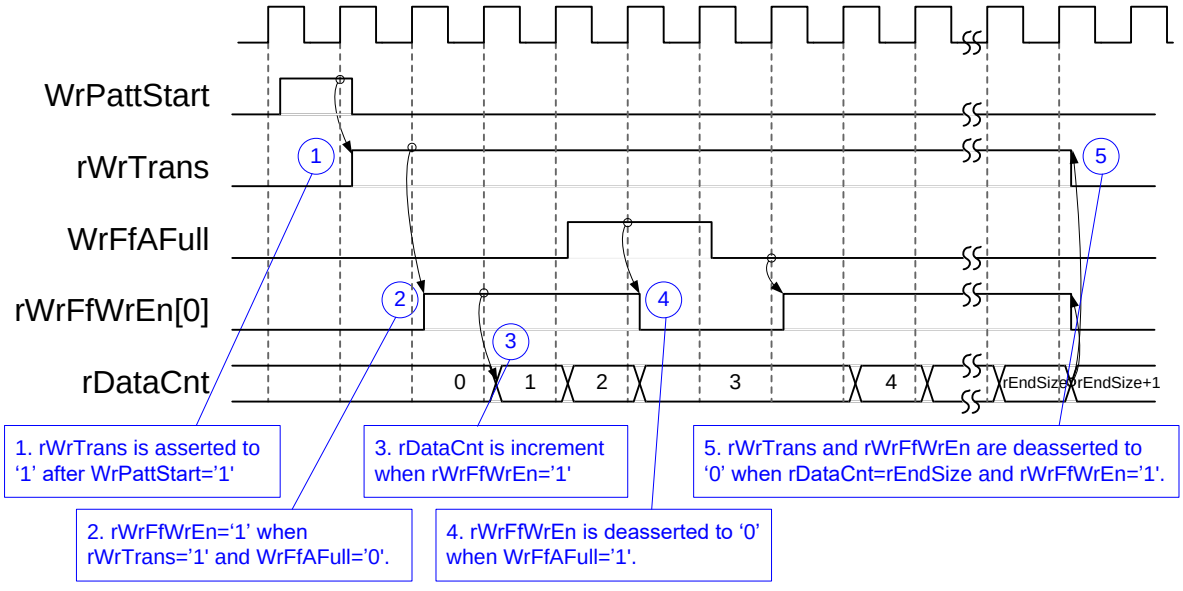

Figure 2-3 Timing diagram of Write operation in TestGen

To start Write operation, rWrTrans is asserted to ‘1’ when WrPattStart from LAxi2Reg is asserted to ‘1’ for one clock cycle. After that, rWrFfWrEn[0] is asserted to ‘1’ to send test data to WrFf when rWrTrans=’1’ and WrFfAFull=’0’. If WrFfAFull=’1’, rWrFfWrEn[0] will be de-asserted to ‘0’ to pause data transferring. rDataCnt is data counter to check total transfer size, increased by rWrFfWrEn[0]. When total data are transferred completely (rDataCnt=rEndSize), rWrTrans and rWrFfWrEn[0] are de-asserted to ‘0’ to stop data transferring. rEndSize is total transfer size in sector unit which is calculated by TrnLen signal.

For Read operation, RdFfRdEn is designed by connecting NOT logic to RdFfEmpty. rDataCnt is increased when RdFfRdEn is asserted to ‘1’. In Read operation, rDataCnt is used to generate test pattern to verify with the received data (RdFfRdData).

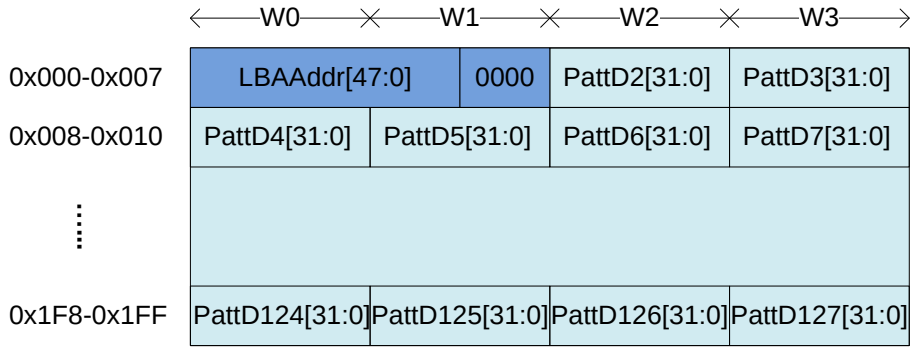

Figure 2-4 Test pattern format in each sector

Block no.1 in lower side of Figure 2-2 shows the logic for generating test pattern in TestGen module. To create unique test data for each sector, test pattern is designed as shown in Figure 2-4.

Test pattern consists of two parts, i.e. 64-bit header in word#0 and word#1 of each sector and test data in word#2 – word#127. 64-bit header is created by using LBA address value of the data (LBA address is the address in sector unit). As shown in Figure 2-2, TrnAddr is loaded to be initial value of rTrnAddr. rTrnAddr is applied to be 64-bit header of each sector and increased after completing to transfer one sector data. rDataCnt and write/read enable signal are monitored to check end of sector transferring.

TestGen supports to generate five patterns, i.e. 32-bit increment, 32-bit decrement, all 0, all 1, and 32-bit LFSR. 32-bit increment is generated by using lower-bit of rTrnAddr and rDataCnt. Decrement pattern is designed by using NOT logic with increment data. 32-bit LFSR counter uses the 1st DW data (LBA[31:0]) to be initial value for generating test pattern in each sector. The equation of LFSR is x^31 + x^21 + x + 1.

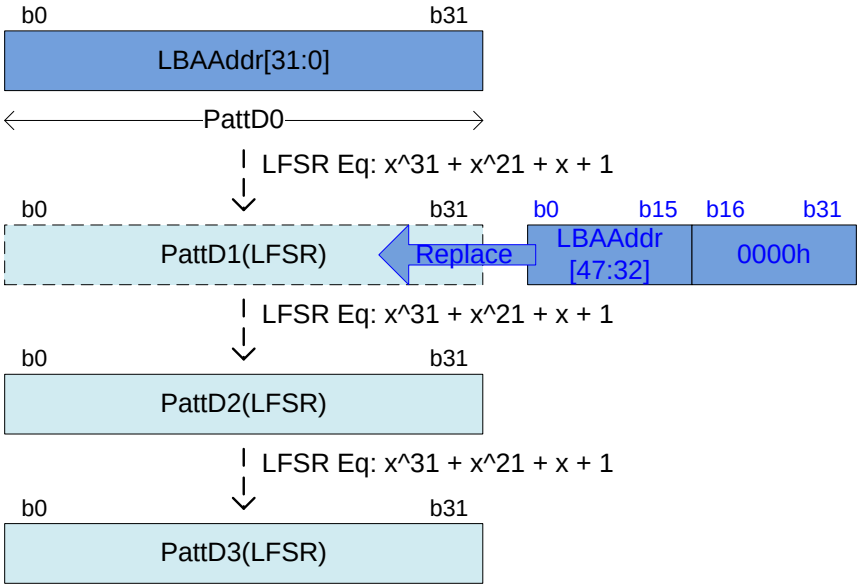

Figure 2-5 LFSR pattern in TestGen

3-bit PattSel signal are used to select one of five test patterns. Header Inserter logic inserts 64-bit header to be the 1st and 2nd data of each sector. After that, test data from pattern counter is transferred to rWrFfWrData. In Read command, rWrFfWrData is used to be expected value to compare with read data from FIFO (RdFfRdData). PattFail is asserted to '1' when data verification is failed.

2.2 SATA

User interface of HCTL-IP is designed by using dgIF typeS format. CMD interface is connected to LAxi2Reg to receive the parameter from user through Serial console. 32-bit data bus is connected with U2IPFIFO and IP2UFIFO. To transfer data with SATA device, HCTL-IP is connected to SATA-IP and SATA PHY, as shown in Figure 2-6.

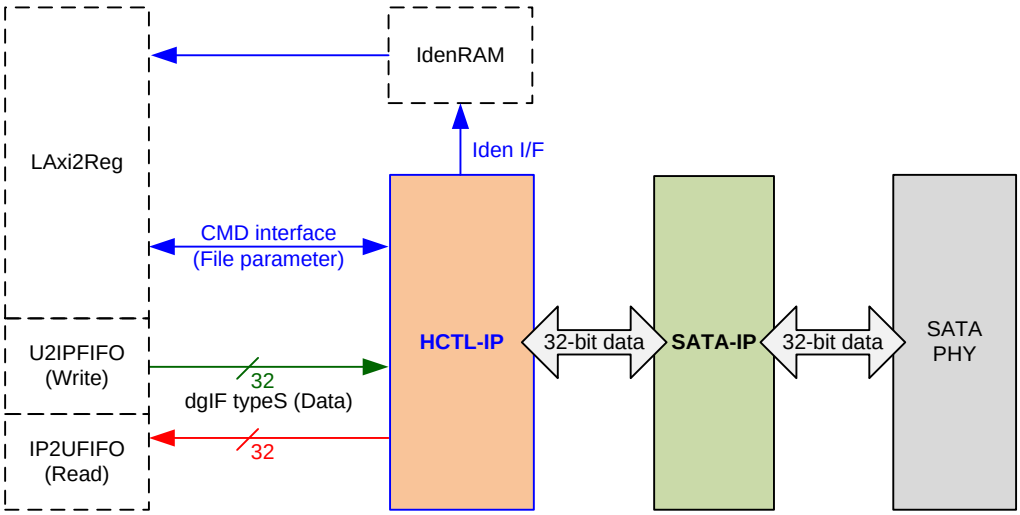

Figure 2-6 SATA hardware

2.2.1 HCTL-IP

HCTL-IP implements application layer of SATA protocol to create and decode SATA FIS packet with SATA-IP. Four commands are supported, i.e. Identify device, Security erase, Write, and Read command. More details of HCTL-IP are described in datasheet.
http://www.dgway.com/products/IP/SATA-IP/dg_sata_host_ip_data_sheet_en.pdf

2.2.2 SATA-IP

SATA-IP implements some parts of transport layer and link layer of SATA protocol. It includes two asynchronous FIFOs to support different clock domain of user interface and PHY interface. More details of SATA-IP are described in datasheet.
http://www.dgway.com/products/IP/SATA-IP/dg_sata_ip_data_sheet_7series_en.pdf

2.2.3 SATA PHY

This module is the example HDL logic which implements state machine for controlling OOB (Out-of-band) sequence with SATA device. The logic includes Xilinx transceiver which is different in each FPGA model. More details of SATA PHY are described in SATA-IP reference design.

2.3 CPU and Peripherals

The hardware is connected to CPU through AXI4-Lite bus, similar to other CPU peripherals. The hardware registers are mapped to CPU memory address, as shown in Table 2-1. The control and status registers for CPU access are designed in LAxi2Reg.

LAxi2Reg connects to many hardwares in the system such as TestGen, HCTL-IP, and IdenRAM to get the control and status signals of each module. As shown in Figure 2-7, there are two clock domains applied in this block, i.e. CpuClk which is used to interface with AXI4-Lite bus of CPU and UserClk which is user clock domain for TestGen and HCTL-IP.

AsyncAxiReg includes asynchronous circuit between CpuClk and UserClk. More details of each hardware are described as follows.

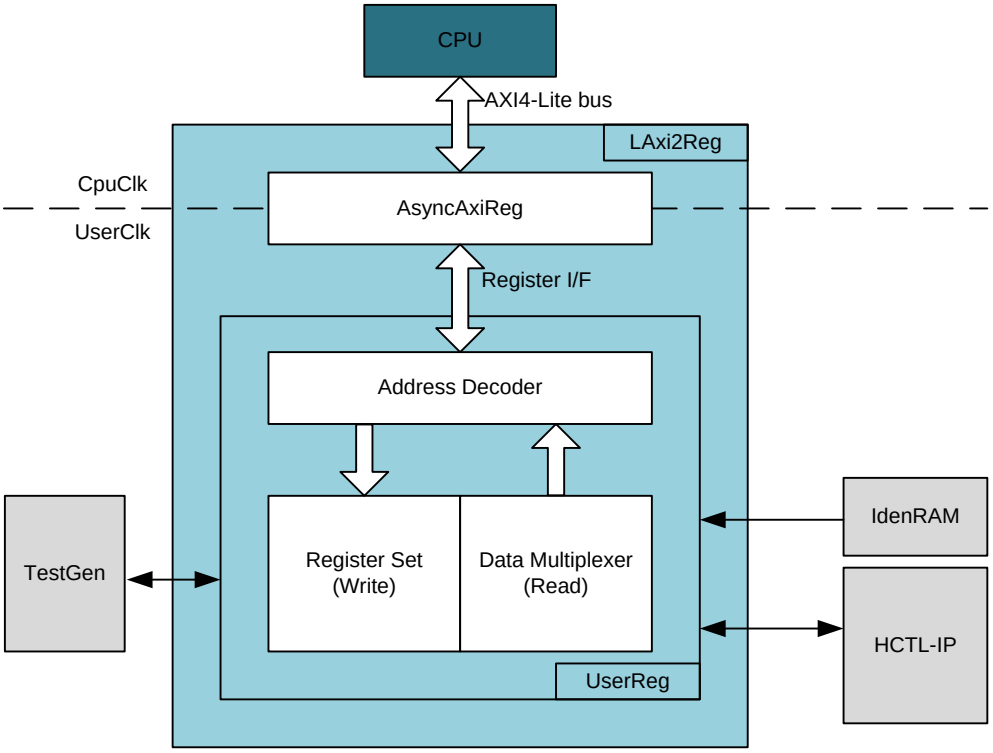

Figure 2-7 CPU and peripherals hardware

2.3.1 AsyncAxiReg

This module is designed to convert the signal interface of AXI4-Lite to be register interface. Also, it supports to convert clock domain from CpuClk to be UserClk domain. Timing diagram of register interface is shown in Figure 2-8.

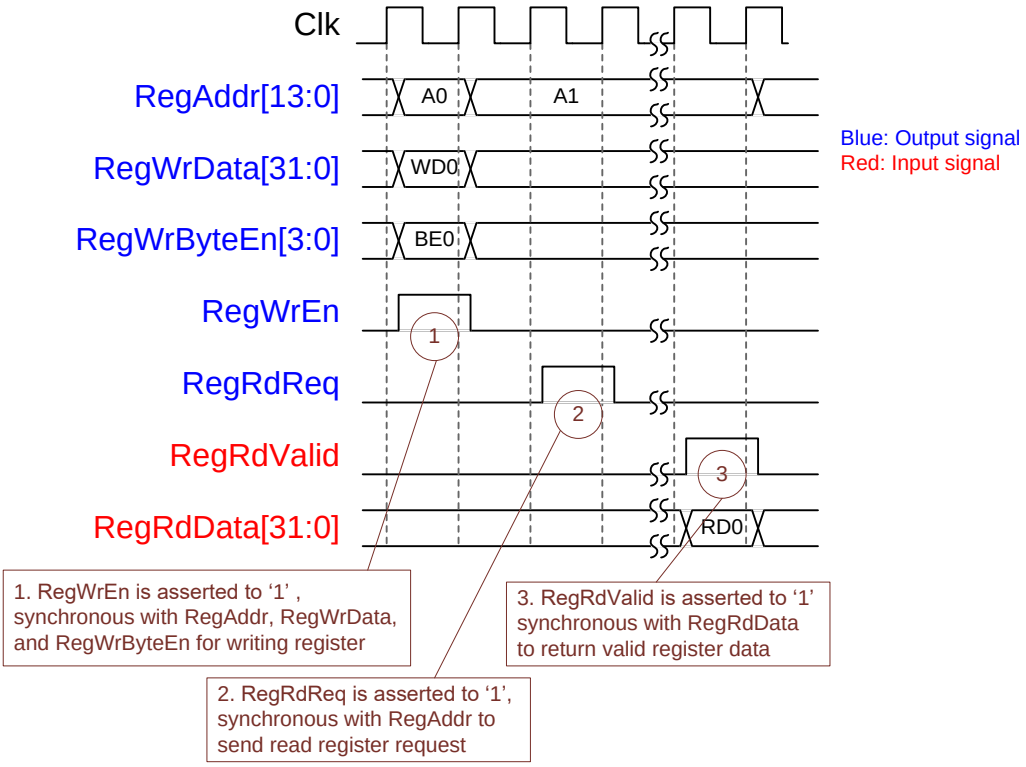

Figure 2-8 Register interface timing diagram

To write register, timing diagram is same as RAM interface. RegWrEn is asserted to '1' with the valid signal of RegAddr (Register address in 32-bit unit), RegWrData (write data of the register), and RegWrByteEn (the byte enable of this access: bit[0] is write enable for RegWrData[7:0], bit[1] is used for RegWrData[15:8], …, and bit[3] is used for RegWrData[31:24]).

To read register, AsyncAxiReg asserts RegRdReq='1' with the valid value of RegAddr (the register address in 32-bit unit). After that, the module waits until RegRdValid is asserted to '1' to get the read data through RegRdData signal. During read access, RegAddr holds the same value until RegRdValid is asserted to '1'.

2.3.2 UserReg

The details of UserReg module is shown in Figure 2-7. After RegWrEn or RegRdReq is asserted to ‘1’ by AsyncAxiReg to request write or read register access, RegAddr is read by Address decoder to select the active register. For write access, RegWrData signal is loaded to be the new value for the requested register. In this module, RegWrByteEn is not used, so CPU firmware needs to access the hardware register by using 32-bit pointer only.

For read request, there are many status signals for CPU access from TestGen, HCTL-IP, and IdenRAM. So, data multiplexer with pipelines register are designed to select the read data to return to CPU following RegAddr value. RegRdValid is designed by using one D Flip-flop, input by RegRdReq signal. So, the read access has one clock cycle latency, measured by the delay time from RegRdReq to RegRdValid.

Memory map of control and status signals inside UserReg module is shown in Table 2-1.

Table 2-1 Register Map

Address Rd/Wr	Register Name (Label in the “hsataiptest.c”)	Description
BA+0x00 Wr	User Address (Low) Reg (USRADRL_REG)	[31:0]: Input to be start sector address (UserAddr[31:0] of dgIF typeS for HCTL-IP)
BA+0x04 Wr	User Address (High) Reg (USRADRH_REG)	[15:0]: Input to be start sector address (UserAddr[47:32] of dgIF typeS for HCTL-IP)
BA+0x08 Wr	User Length (Low) Reg (USRLENL_REG)	[31:0]: Input to be transfer length in sector unit (UserLen[31:0] of dgIF typeS for HCTL-IP)
BA+0x0C Wr	User Length (High) Reg (USRLENH_REG)	[15:0]: Input to be transfer length in sector unit (UserLen[47:32] of dgIF typeS for HCTL-IP)
BA+0x10 Wr	User Command Reg (USRCMD_REG)	[1:0]: Input to be user command (UserCmd of dgIF typeS for HCTL-IP) “00”-Identify device, “01”-Security erase, “10”-Write Dev, “11”-Read Dev. When this register is written, the design generates command request to HCTL-IP to start new command operation.
BA+0x14 Wr	Test Pattern Reg (PATTSEL_REG)	[2:0]: Test pattern select “000”-Increment, “001”-Decrement, “010”-All 0, “011”-All 1, “100”-LFSR
BA+0x100 Rd	User Status Reg (USRSTS_REG)	[0]: UserBusy of dgIF typeS for HCTL-IP (‘0’: Idle, ‘1’: Busy) [1]: UserError of dgIF typeS for HCTL-IP (‘0’: Normal, ‘1’: Error) [2]: Data verification fail (‘0’: Normal, ‘1’: Error) [4:3]: SATA speed from IP “00”: No linkup, “01”: SATA Gen1 (Not supported for all designs), “10”: SATA Gen2 (Not supported for KCU105), “11”: SATA Gen3
BA+0x104 Rd	Total disk size (Low) Reg (LBASIZEL_REG)	[31:0]: Total capacity of SATA device in sector unit (LBASize[31:0] of dgIF typeS for HCTL-IP)
BA+0x108 Rd	Total disk size (High) Reg (LBASIZEH_REG)	[15:0]: Total capacity of SATA device in sector unit (LBASize[47:32] of dgIF typeS for HCTL-IP)
BA+0x10C Rd	User Error Type Reg (USRERRTYPE_REG)	[31:0]: User error status, directly mapped from UserErrorType[31:0] of HCTL-IP.
BA+0x11C Rd	SATA Host IP Test pin Reg (TESTPIN_REG)	[31:0]: TestPin[31:0] which is directly mapped from HCTL-IP
BA+0x120 Rd	Data Failure Address (Low) Reg (RDFAILNOL_REG)	[31:0]: Latch value of failure address[31:0] in byte unit from read command
BA+0x124 Rd	Data Failure Address (High) Reg (RDFAILNOH_REG)	[24:0]: Latch value of failure address [56:32] in byte unit from read command
BA+0x130 Rd	Expected value Word0 Reg (EXPPATW0_REG)	[31:0]: Latch value of expected data [31:0] from read command
BA+0x140 Rd	Read value Word0 Reg (RDPATW0_REG)	[31:0]: Latch value of read data [31:0] from read command
BA+0x150 Rd	Current test byte (Low) Reg (CURTESTSIZEL_REG)	[31:0]: Current test data size of TestGen module in byte unit (bit[31:0])
BA+0x154 Rd	Current test byte (High) Reg (CURTESTSIZEH_REG)	[24:0]: Current test data size of TestGen module in byte unit (bit[56:32])
BA+0x2000 – 0x21FF	Identify Device Command Data (IDENCTRL_REG)	512-byte Identify Device Data

3 CPU Firmware

After system boot-up, CPU initializes its peripherals such as UART and Timer. Next, CPU waits HCTL-IP completes initialization process (USRSTS_REG[0]='0'). After that, main menu is displayed. CPU firmware supports four commands following USRCMD_REG value, i.e. "00" for Identify device, "01" for Security erase, "10" for Write, and "11" for Read. More details of the sequence in each command are described as follows.

3.1 Identify device command

The sequence of the firmware when user selects Identify device command is below.

1) Set USRCMD_REG="00". Next, Test logic sends Identify device command to HCTL-IP. HCTL-IP busy flag (USRSTS_REG[0]) changes from '0' to '1'.
2) CPU waits until the operation is completed or some errors are found by monitoring USRSTS_REG value. Bit[0] is de-asserted to '0' when command is completed. Bit[1] is asserted to '1' when some errors are detected. In case of error condition, there is error message displayed on the console. If the command is completed, the data from Identify device command of SATA device will be stored in IdenRAM.
3) CPU reads Identify device data from IdenRAM which is mapped to IDENCTRL_REG address. Then, CPU displays the information such as SATA device model name, security feature set supported, and erase time value on Serial console. Also, device capacity (LBASIZEL/H_REG) is displayed in GB unit on the console.

3.2 Write/Read command

The sequence of the firmware when user selects Write/Read command is below.

1) Receive start address, transfer length, test pattern through Serial console. If some inputs are invalid, the operation will be cancelled.
2) Get all inputs and set the value to USRADRL/H_REG, USRLENL/H_REG, PATTSEL_REG, and USRCMD_REG (USRCMD_REG="10" for write transfer, and "11" for read transfer).
3) CPU waits until the operation is completed or some errors (except verification error) are found by monitoring USRSTS_REG[2:0].
   If USRSTS_REG[2] is asserted to '1', verification error message will be displayed. After that, CPU still runs until end of operation or user inputs any key to cancel operation.
4) During running command, current transfer size reading from CURTESTSIZE_REG is displayed every second. Finally, test performance is displayed on Serial console when command is completed.

3.3 Security erase command

The sequence of the firmware when user selects Security erase command is below.

1) Set USRCMD_REG="01". Next, test logic sends Security erase command to HCTL-IP. HCTL-IP busy flag (USRSTS_REG[0]) changes from '0' to '1'.
2) CPU waits until command complete by monitoring USRSTS_REG value. Bit[0] is de-asserted to '0' when command is completed. This operation may use long time to operate, so there is dummy message displayed on the console every second to show system alive status. Finally, total time usage is displayed on Serial console when command is completed.

4 Example Test Result

The example test result when running demo system by using 256 GB Samsung 850 Pro is shown in Figure 4-1.

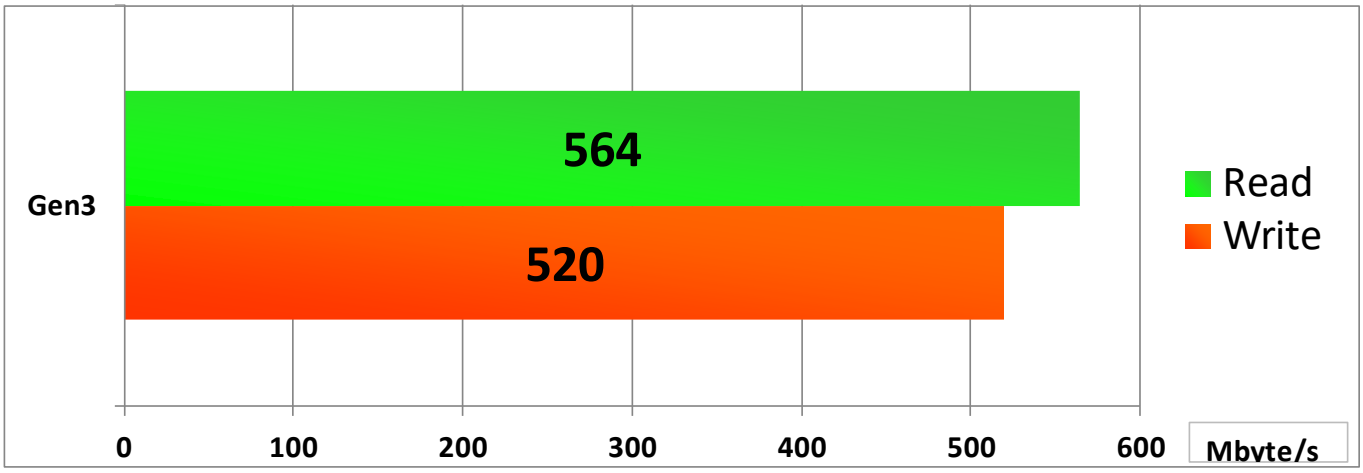

Figure 4-1 Test Performance of HCTL-IP demo by using Samsung 850 Pro SSD

By using SATA Gen3 on KCU105 board, write performance is about 520 Mbyte/sec and read performance is about 564 Mbyte/sec.

5 Revision History

Revision	Date	Description
1.0	13-Oct-14	Initial Release
1.1	29-Aug-16	Add CPU system for Serial console interface
1.2	9-Nov-16	Add Security erase command
1.3	30-Jan-17	Update signal to dgIF typeS
1.4	1-Aug-17	Add LFSR pattern
1.5	11-May-17	Update the details of reference design

Copyright: 2014 Design Gateway Co,Ltd.