SATA-IP Device reference design on KC705 manual

Rev1.0 20-Jun-13

1. Introduction

Serial ATA (SATA) is an evolutionary replacement for the Parallel ATA (PATA) physical storage interface. SATA interface increases speed transfer to be 3.0 Gbps for SATA-II, and 6.0 Gbps for SATA-III. To communication by SATA protocol, there are four layers in its architecture, i.e. Application, Transport, Link, and Phy.

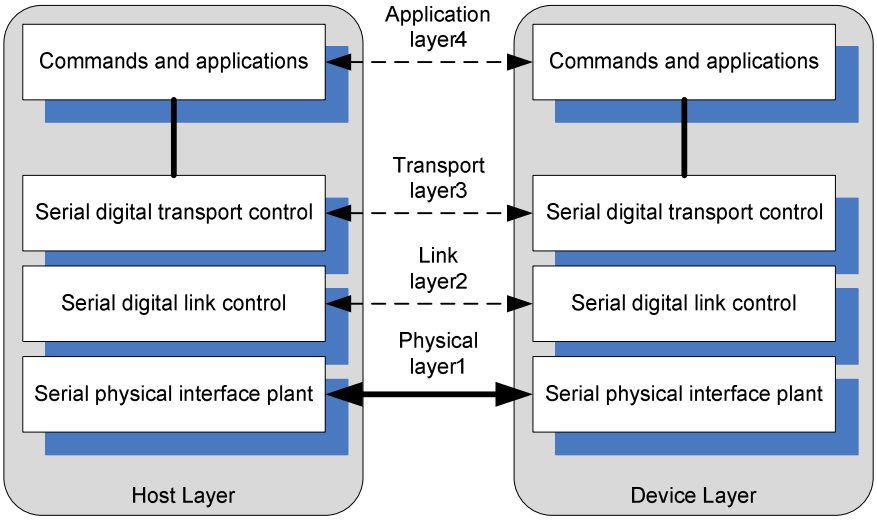

Figure 1 SATA Communication Layer

The Application layer is responsible for overall ATA command execution, including controlling Command Block Register accessed. The Transport layer is responsible for placing control information and data to be transferred between the host and device in a packet/frame, known as a Frame Information Structure (FIS). The Link layer is responsible for taking data from the constructed frames, encoding or decoding each byte using 8b/10b, and inserting control characters such that the 10-bit stream of data may be decoded correctly. The Physical layer is responsible for transmitting and receiving the encoded information as a serial data stream on the wire.

This reference design provides evaluation system which implements all SATA communication layers for Device side to transfer high speed data with SATA-III Host PC. The SATA-IP core is designed to operate with GTX transceiver of the Kintex-7 platform in the reference design on KC705 Evaluation board. More details are described as follows.

2. Environment

This reference design is based on the following environment as shown in Figure2.

- KC705 Platform
- Vivado2012.4 and SDK14.4
- SATA connector on AB07-USB3FMC, provided by Design Gateway
- SATA cross over cable or SATA standard cable with AB02-CROSSOVER, provided by Design Gateway
- SATA-III Host PC
- USB Micro-B cable for FPGA configuration
- USB Mini-B cable for serial communication. For serial communication, set baud rate=115,200 / data=8bit / Non-Parity / Stop=1bit.

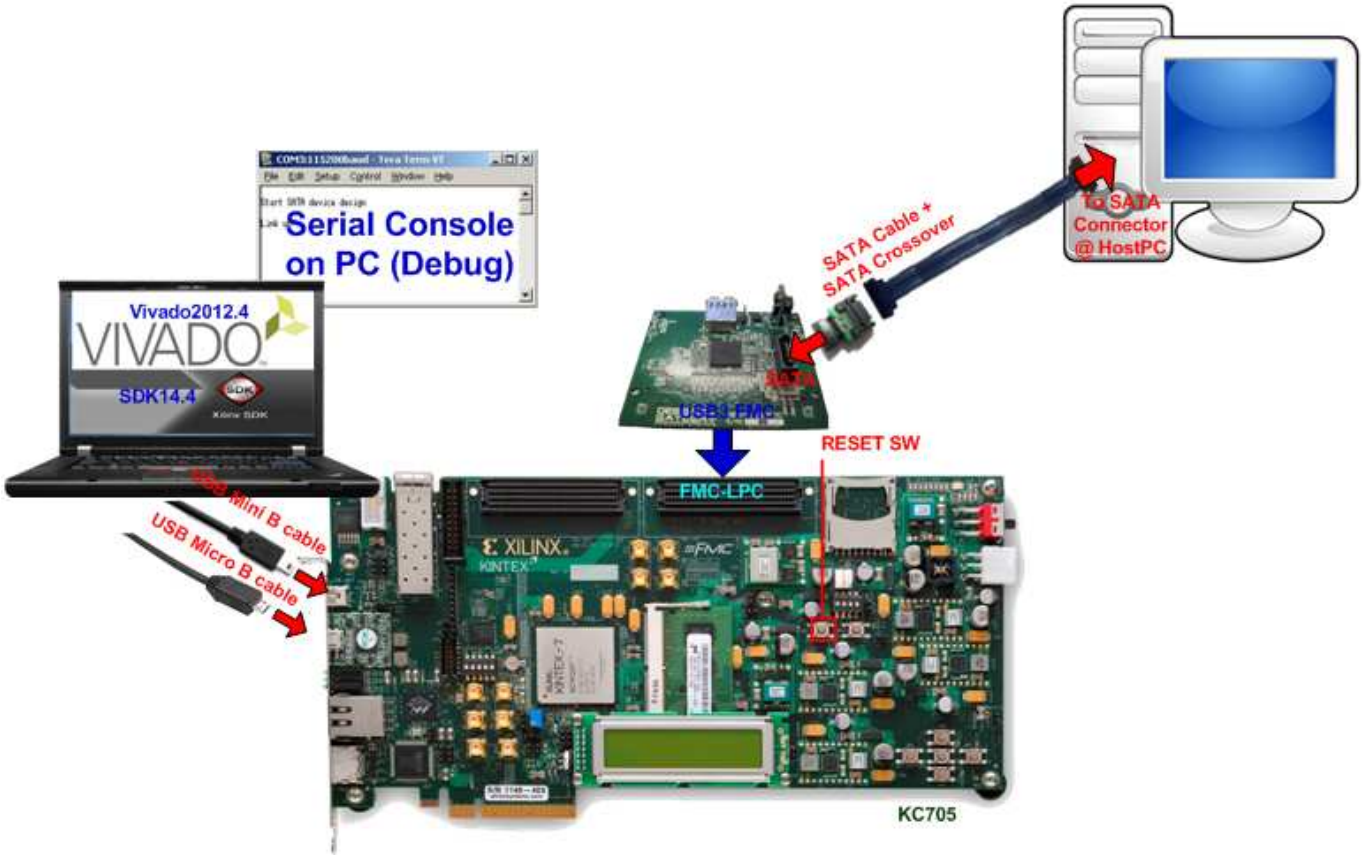

Figure 2 Reference design environment

Refer to “SATA-IP Device Demo Instruction on KC705” for operation procedure of this reference design. For evaluation version, the system includes 1-hour limitation to use. After timeout, the system will stop any data transfer.

3. Hardware description

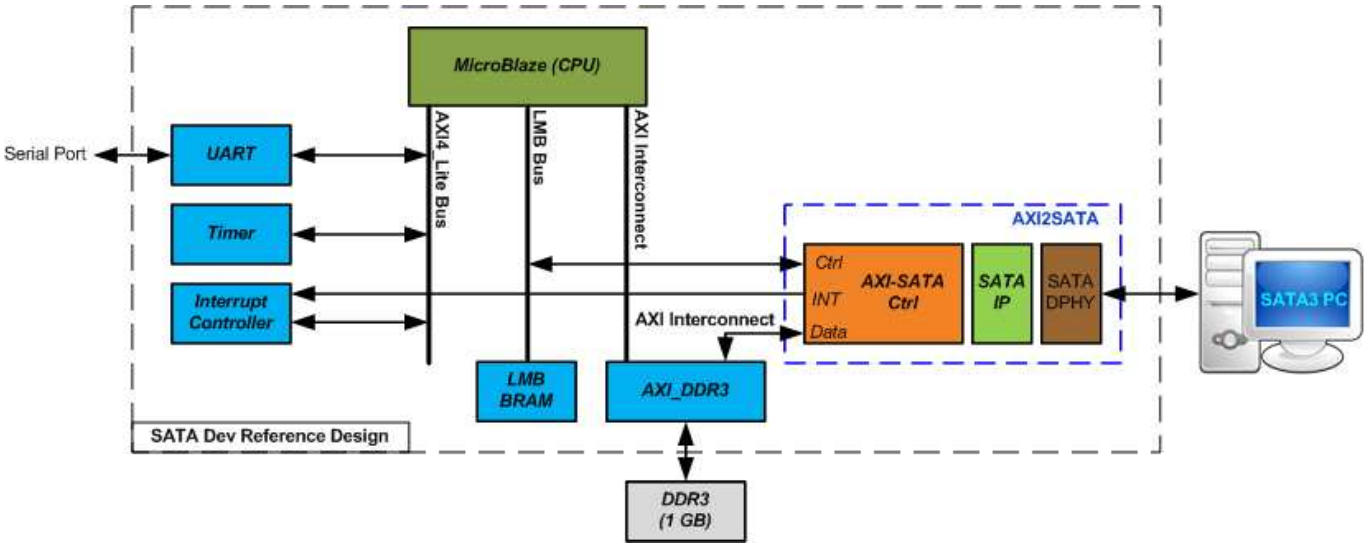

Figure 3 Block diagram of the reference design

- SATA IP Device design implementation on Kintex7 FPGA

The demo is designed based on EDK system. MicroBlaze and other hardwares operate to emulate DDR3 memory on board to be RAMDISK for SATA3 PC access as SATA peripheral. Some part of DDR3 memory is reserved to be temp buffer for storing SATA FIS packet transferred between MicroBlaze and SATA-IP.

SATA-IP can plug-in to EDK system by using two bus interfaces, i.e. AXI4 Interface for data bus and LMB interface for control bus. The demo provides the example module to convert SATA-IP interface to both buses. User can modify this module for user system. The system also includes UART interface for debugging, timer for checking timeout, and interrupt controller for interrupt when end of sending/receiving SATA packet with PC.

MicroBlaze firmware is responsible for SATA Application layer, as shown in Figure 4. It will decode the command from PC and response SATA FIS packet back to PC. The other SATA layers are designed by AXI2SATA and SATAPHY module. MicroBlaze firmware is stored into BRAM through LMB Bus interface, same bus with AXI2SATA control signal. More details about the hardware design are follows.

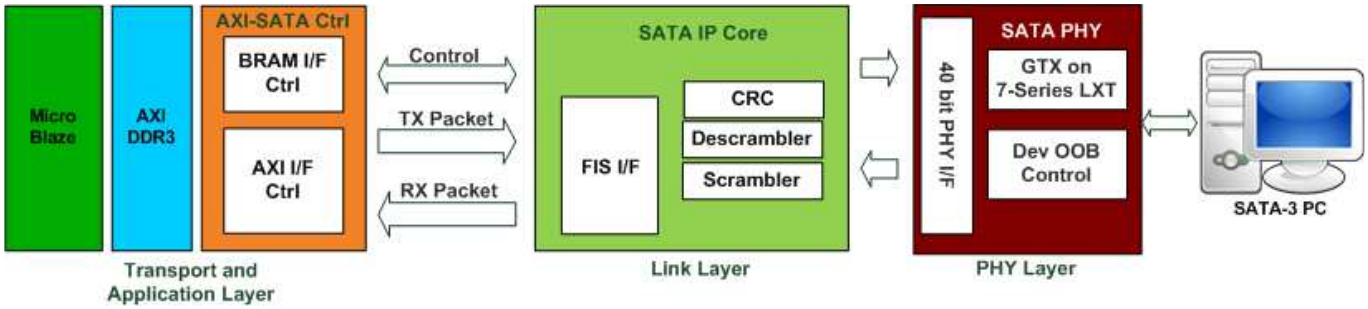

Figure 4 AXI2SATA Block Diagram

- PHY Layer

This layer is designed by using GTX module (built-in high speed serial circuit) of Kintex-7 LXT device, operating with logic control to generate OOB sequence and initialization sequence of physical layer. The OOB sequence on device demo is reversed from host demo. State machine to control OOB is designed in "dev_oob_control.vhd" which is sub-module of "sata2dphy_k7.vhd", the top layer of Device PHY source code. PHY design is fixed into SATA3 speed.

The reference design follows PLL and GTX reset sequence described in "7 Series FPGAs GTX/GTH Transceivers" user guide, refer to "Reset and Initialization" section in ug476 for detailed reset sequence.

Before building user board, user must read carefully and must follow design guide line described in UG476 (7 Series FPGAs GTX/GTH Transceivers User Guide).

- Link Layer by SATA-IP

Link Layer and some part of Transport layer are implemented by SATA-IP. FIS packet format is converted to lower layer by including CRC and scramble processing. Also, packet from physical layer is decoded and arrange to FIS packet format to interface with user. More details about SATA-IP interface are described in "dg_sata_ip_datasheet_kt7_en" document.

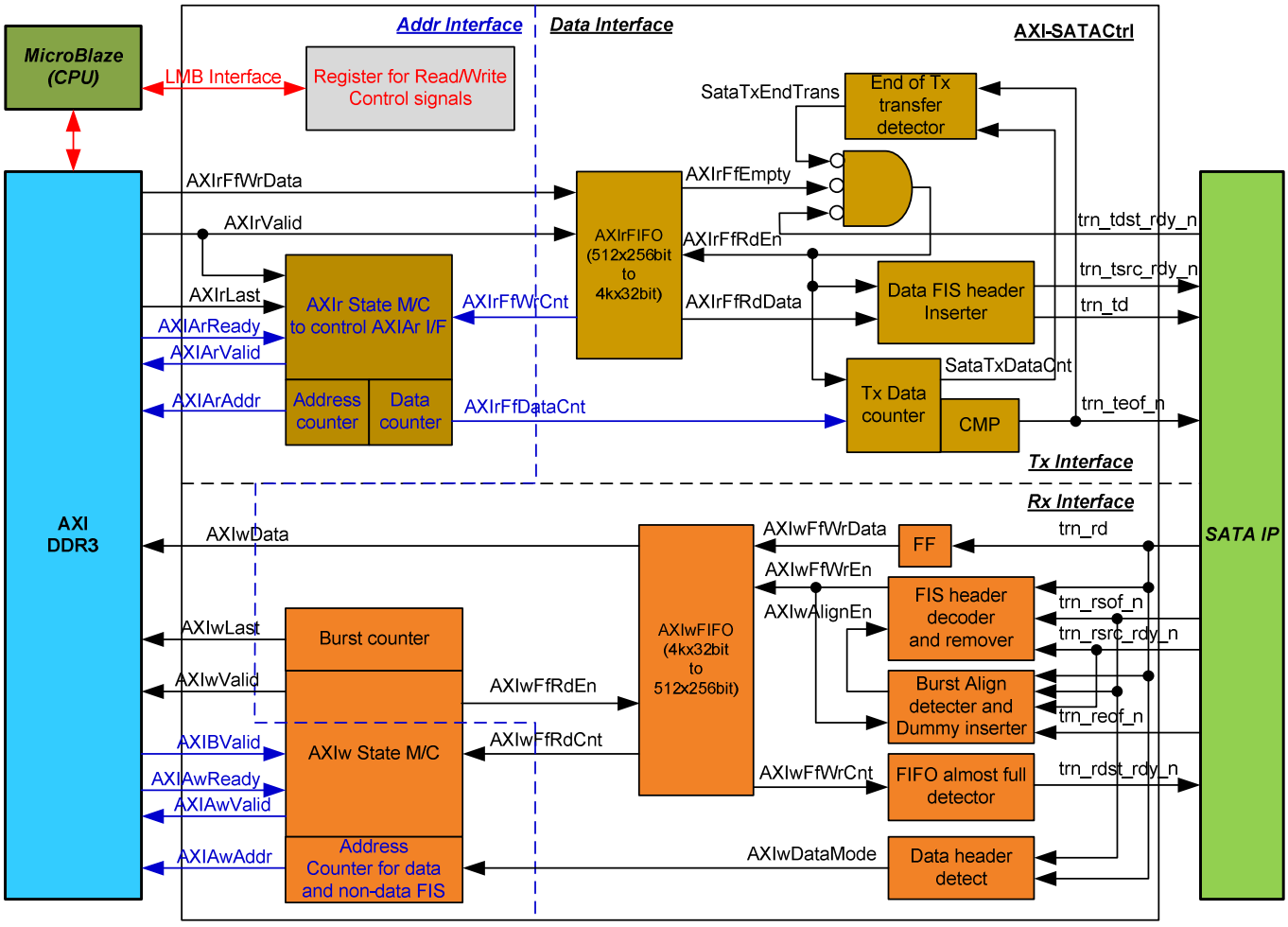

Figure 5 Block diagram of AXI-SATACtrl

- Transport Layer by AXI-SATACtrl

AXI-SATACtrl is the logic design to connect SATA-IP interface to standard interface within EDK system, i.e. LMB interface for control signal and AXI4 for data signal. LMB interface connects to MicroBlaze while AXI4 interface running as master mode connects to DDR3 controller.

Register map of control signals to interface with MicroBlaze by LMB bus is shown in Table 1. Main operation of these registers is to define DDR address for transferring FIS data, transfer length, and FIS type (data or non-data). The registers are read by MicroBlaze to check AXI-SATACtrl status. The non-data FIS will be stored in reserved area of DDR while the data FIS will be stored in RAMDISK area of DDR, as shown in Figure 6.

The reserved area to store non-data FIS for both TX and RX direction is defined to the first 256 MB of DDR3, and the remaining 768 MB of DDR3 is defined to be RAMDISK area. To improve performance, the frequently used TX FIS such as Status FIS, DMA Activate FIS are permanent created and stored in DDR3, so MicroBlaze can create this FIS sending to HostPC without re-generate the FIS every time.

Address Rd/Wr	Register Name (Label in the “sata_host.c”)	Description (Bit order is little endian)
BA+0x04 Rd	Error Code Reg. (ERROR_CODE)	SATA IP Error code after transmit/receive completion to detect CRC or FIS error.
BA+0x0C Rd	Receive Word Count Reg. (RX_COUNT)	[31] : Received FIS type in this interrupt (‘1’: Non-Data FIS, ‘0’: Data FIS). This bit is cleared by writing bit[29] of INT_CLEAR = ‘1’. [23:0] : Total Received word count of FIS data. Auto clear when next transfer start (CONTROL Reg is written).
BA+0x00 Wr	Transmit Data Address Reg. (TX_ADDR)	Set DDR3 start address of transmit FIS data area Bit[8:0] of this value needs to be equal to 0.
BA+0x04 Wr	Received Data Address1 Reg. (RX_ADDR)	Set DDR3 start address of received other FIS area (except DATA FIS type). Bit[8:0] of this value needs to be equal to 0.
BA+0x08 Wr	Control Reg. (CONTROL)	[31] : Hardware Reset [30] : Start Transmit data [29] : FIS type (‘1’: Data, ‘0’: Others) [15:0] : Total Transmit word count. RX_COUNT register is cleared by write operation to this register
BA+0x0C Wr	Received Data Address2 Reg. (RX2_ADDR)	Set DDR3 start address of received DATA FIS area Bit[8:0] of this value needs to be equal to 0.
BA+0x10 Wr	Interrupt Clear Reg (INT_CLEAR)	[31] : Set this bit to clear ip2host interrupt [30] : Set this bit to clear host2ip interrupt [29] : Set this bit to clear received FIS type

Table 1 Register mapping from CPU side

(BA : Base Address)

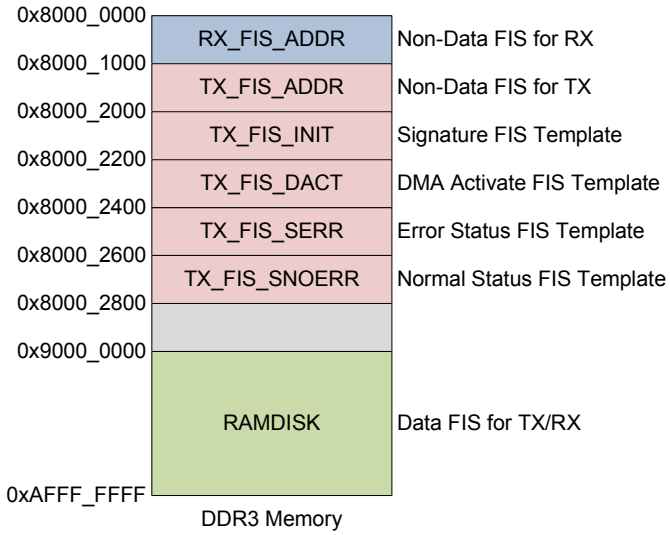

Figure 6 DDR3 Memory Map

To transfer data by AXI interface, signals are divided into four groups, i.e. AXIAr for addressing request of receiving direction, AXIr for data transferring of receiving direction, AXIAw for addressing request for transmitting direction, and AXIw for data transferring of transmitting direction. Addressing request of each direction is controlled by state machine. FIFO is applied for data buffering and converting different bus size between 256-bit (AXI bus size) and 32-bit (SATA-IP bus size).

For data receiving mode from DDR3 to SATA3 PC, MicroBlaze sets control register to start transferring data operation. State Machine sends data request to get data from DDR3 to store to AXIrFIFO until total transfer size equal to set value from user. The request will be paused if FIFO is almost full. Read enable signal of AXIrFIFO is simple designed by monitoring Empty flag of FIFO with data ready signal from SATA-IP. Data output from AXIrFIFO will be arranged to FIS packet and Data FIS header will be auto-added to FIS packet by internal logic in case of user setting to send Data FIS type. End-of-packet signal to SATA-IP will be generated when total transfer size to SATA-IP is equal to setting value from user. By above sequence, MicroBlaze can simply create any FIS type with specified FIS size to SATA3 PC through SATA-IP.

For data transmitting mode from SATA3 PC to DDR3, FIS packet from SATA-IP will be decoded by internal logic to check FIS type from FIS header. AXIwDataMode, output signal from the decoder, is fed to address generator for storing Data FIS packet and non-Data FIS packet to DDR at RAMDISK or RX_FIS_ADDR area. Data FIS header will be removed from FIS packet before storing to AXIwFIFO. If data packet storing to AXIwFIFO ends with unaligned burst size, internal logic will add dummy word to FIFO until size aligned. In this demo, burst size to AXI bus is fixed to 256bitx16 beat for high performance throughput. State machine monitors read counter of FIFO that data counter is enough, and then sends request to AXI to transfer data. State machine will go to idle state after completing each burst data transfer. By this sequence, MicroBlaze can monitor and decode received FIS packet by monitoring interrupt signal and total received size from register.

AXI-SATACtrl logic operating with SATA-IP and PHY layer code are stored in “AXI2SATA.vhd”, provided to user as delivery item.

4. Software description

- SATA Device operation

Basically, SATA peripheral Device must support all mandatory commands sent from the Host. But this reference design supports only the minimum command to simplify software so that user can easily understand fundamental SATA Device software operation.

Like Host reference design, communication between the Host and the Device via SATA is done by FIS (Frame Information Structure) data structure. MicroBlaze in the Device design will build FIS data structure on its main memory space, and will send it to the Host by DMA controller that operates bus master. And FIS data sent from the Host is also transferred on the main memory by DMA controller.

MicroBlaze in the SATA Device will operates as below sequence.
(1) After boot-up, send RegD2H FIS to the Host.
(2) Wait command receive.
(3) Execute received command operation.
(4) Send FIS Data
(5) Additional FIS data transmit/receive if necessary.

- Software of reference design description

Software source code of this reference design is stored in “sata_device.c”. As a minimum implementation, this source code has a following limitation.

- There is no optional function support such as S.M.A.R.T (Self-monitoring, Analysis and Reporting Technology)
- 48bit LBA is not supported
- Cache is not supported (Not need because this peripheral is RAMDISK)
- Ultra DMA mode support is Mode 5 or slower.

After Link is established, software sends RegD2H (Device to Host FIS), and then process command sent from the Host. This design prepares original IDENTIFY DEVICE data to limit operation from the Host. Since DDR3 memory capacity on KC705 is 1GBytes and some area needs to be use to store FIS packet with SATA-IP, this software declares only 768MBytes for disk capacity.

To make easy implementation, maximum value of SET MULTIPLE MODE is set to 1 so that READ/WRITE SECTOR command sequence and READ/WRITE MULTIPLE command sequence is identical. In this case, performance is not good. However, in practically there is no problem for typical use because the Host will use UDMA mode rather than legacy PIO mode.

There are two READ/WRITE address modes, i.e. CHS (Cylinder/Head/Sector) mode and LBA (Logical Block Address) mode. Generally, CHS mode is used in the small size HDD access. For this reference design, CHS is converted to be LBA mode access.

The commands implemented in this reference design are READ SECTOR, READ MULTIPLE, WRITE SECTOR, WRITE MULTIPLE, READ DMA, and WRITE DMA. For the other mandatory command, this reference design will do nothing and simply return RegD2H without error.
For other command which is not mandatory command, this reference design will return RegD2H with error.

This reference design implements the minimum command required for simple board operation check, but user should support all the mandatory commands defined in the ATA Standard specification in their application.

5. KC705 real board operation

- OS Boot-up

After Windows7 boot-up completion, user can recognize that Windows7 can detect KC705 as a SATA Peripheral Device and transfer mode is set to Ultra DMA Mode 5, as shown in Figure 7. Some mandatory commands for OS Boot-up are shown in Table 2. The command sequence depends on PC, Device, and OS.

Command code	Command operation
ECh	IDENTIFY DEVICE
EFh	SET FEATURES (Set transfer mode)
C8h	READ DMA
C6h	SET MULTIPLE MODE

Table 2 Command Sample for WIndows7 Boot-up

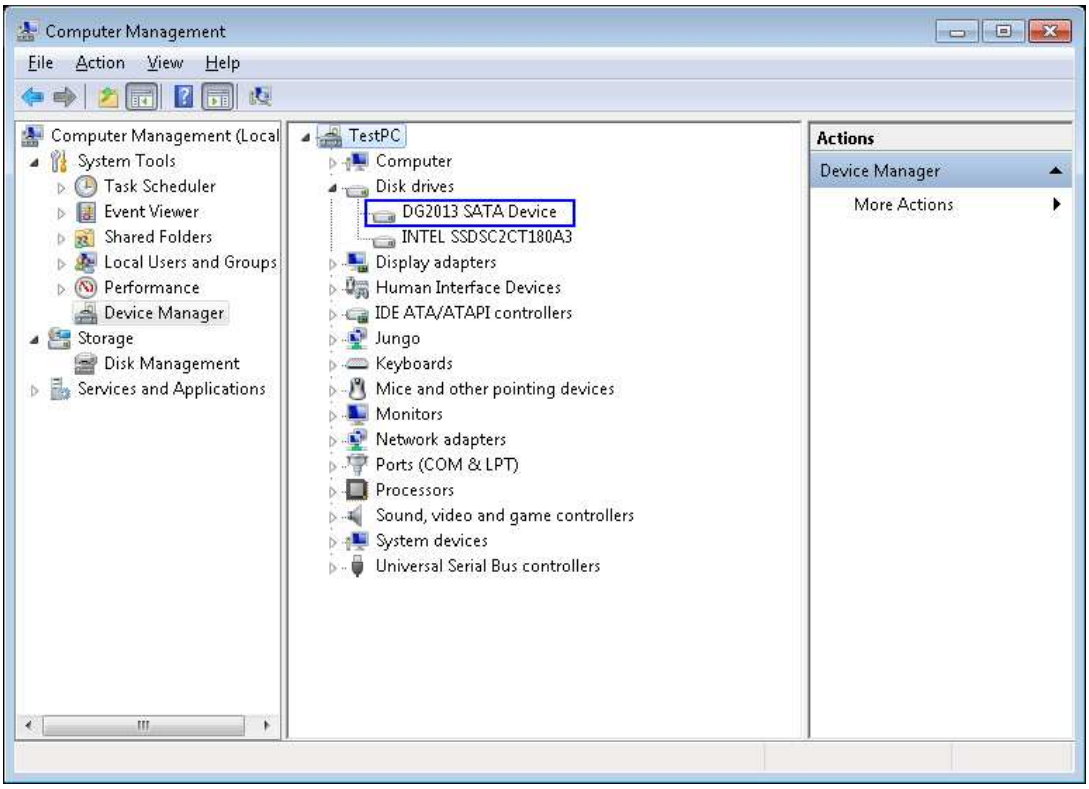

Figure 7 Windows7 detects KC705 as SATA Device

Transfer mode settings will be sent after IDENTIFY DEVICE command. Though this reference design does not support optional S.M.A.R.T command, OS boot-up is successful because this design will send error response of command not support by RegD2H. This design simply returns RegD2H without error when receive IDLE, or IDLE IMMEDIATE command.

Because this design reports maximum value of SET MULTIPLE MODE as 1, Host sets 1 at SET MULTIPLE MODE command.

After Windows7 boot-up completion, user can recognize that Windows7 can detect KC705 as a SATA Peripheral Device and transfer mode is set to Ultra DMA Mode 5.

- OS Format operation

Table 3 shows a command sample for Format operation. Test results are shown in Figure 8.

Command code	Command operation
C8h	READ DMA
CAh	WRITE DMA

Table 3 Command Sample for Format

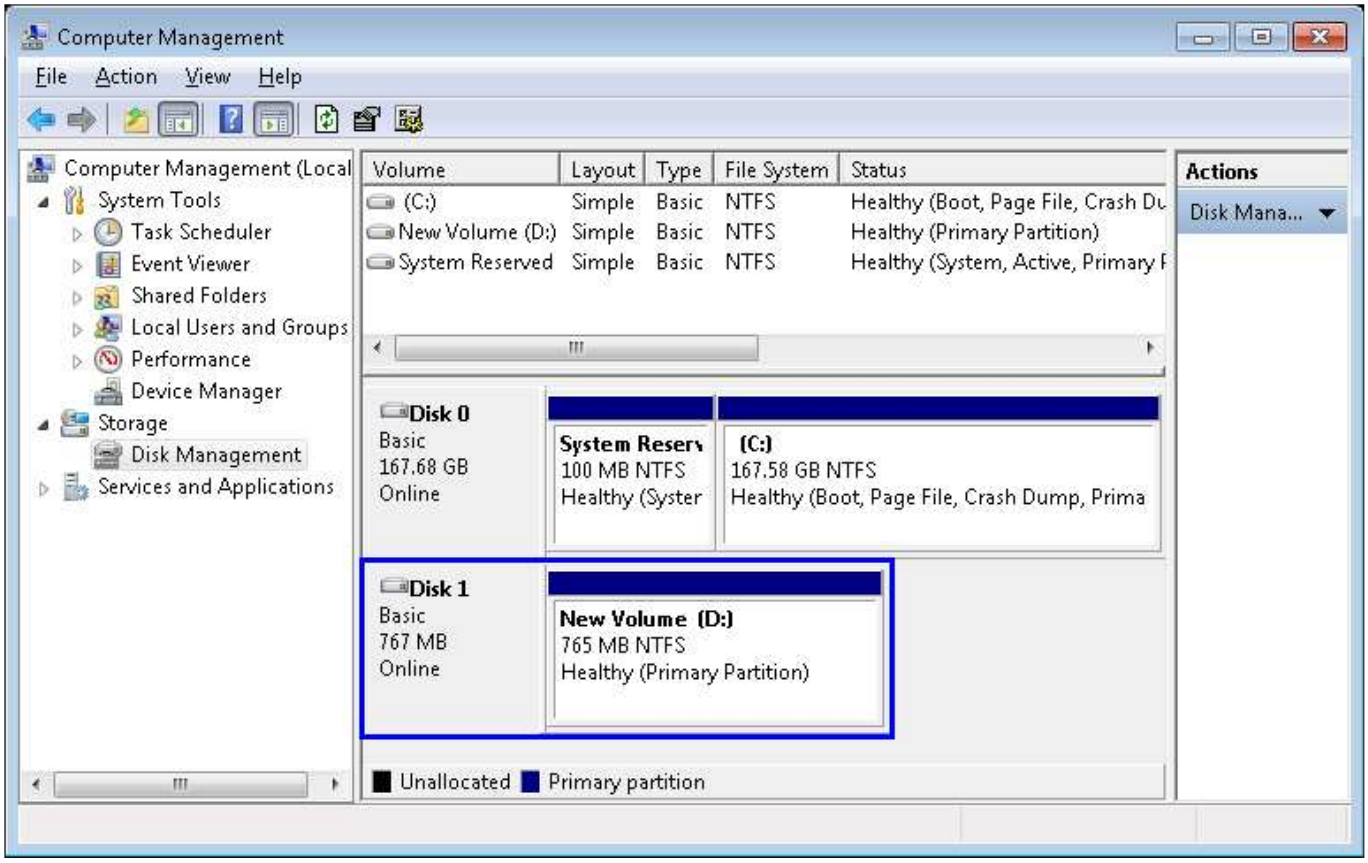

Figure 8 New drive is displayed after Format

Table 4 shows command sample for Windows7 shut down. Windows7 executes remained write data, and then execute FLUSH CACHE command.

Command code	Command operation
CAh	WRITE DMA
E7h	FLUSH CACHE

Table 4 Command Sample for Windows7 shut down

- Performance result

Figure 9 shows test result performance of this reference design on KC705.

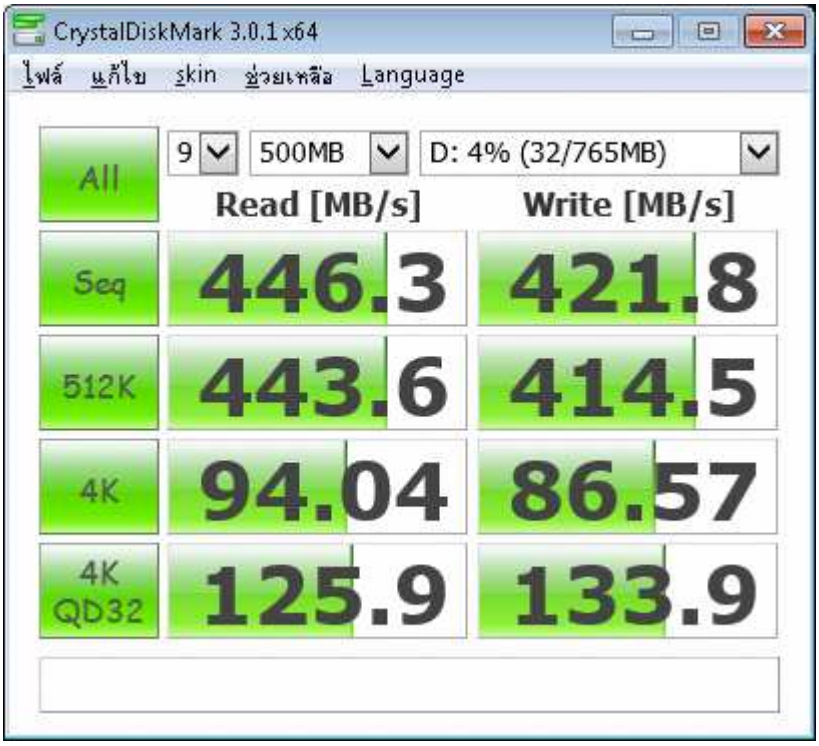

Figure 9 Performance result of read/write data transfer

6. Revision History

Revision	Date	Description
1.0	20-Jun-13	Initial draft