SATA-IP Bridge reference design on KC705 manual

Rev1.0 03-Mar-14

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.

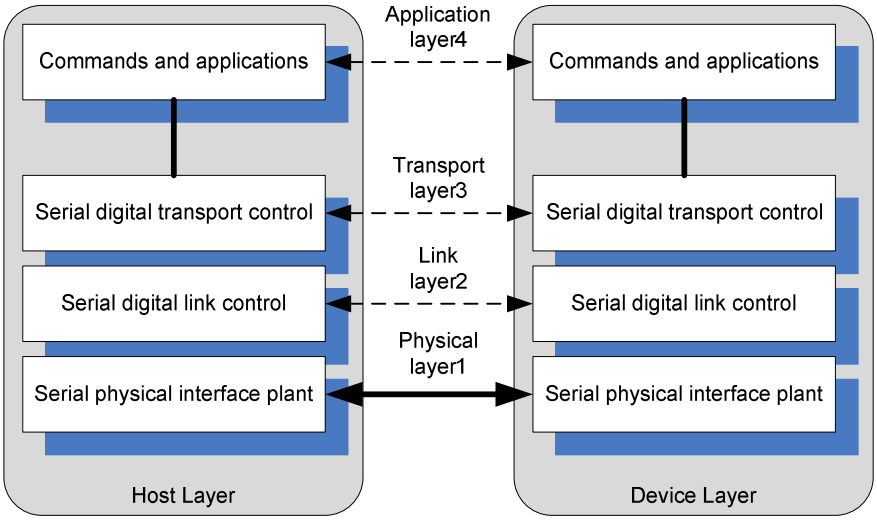

Figure 1 SATA Communication Layer

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 Host and Device side to demonstrate SATA Bridge operation at SATA-III speed. 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
- Vivado 2012.4/SDK14.4
- AB09-FMCRAID board, provided by Design Gateway
- AB02-CROSSOVER board, provided by Design Gateway
- 2.5-inch SATA cable connecting to CN0 connector on FMCRAID board and SATA-III PC
- SATA-III Device (HDD/SSD) connecting to CN1 connector on FMCRAID board
- 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.

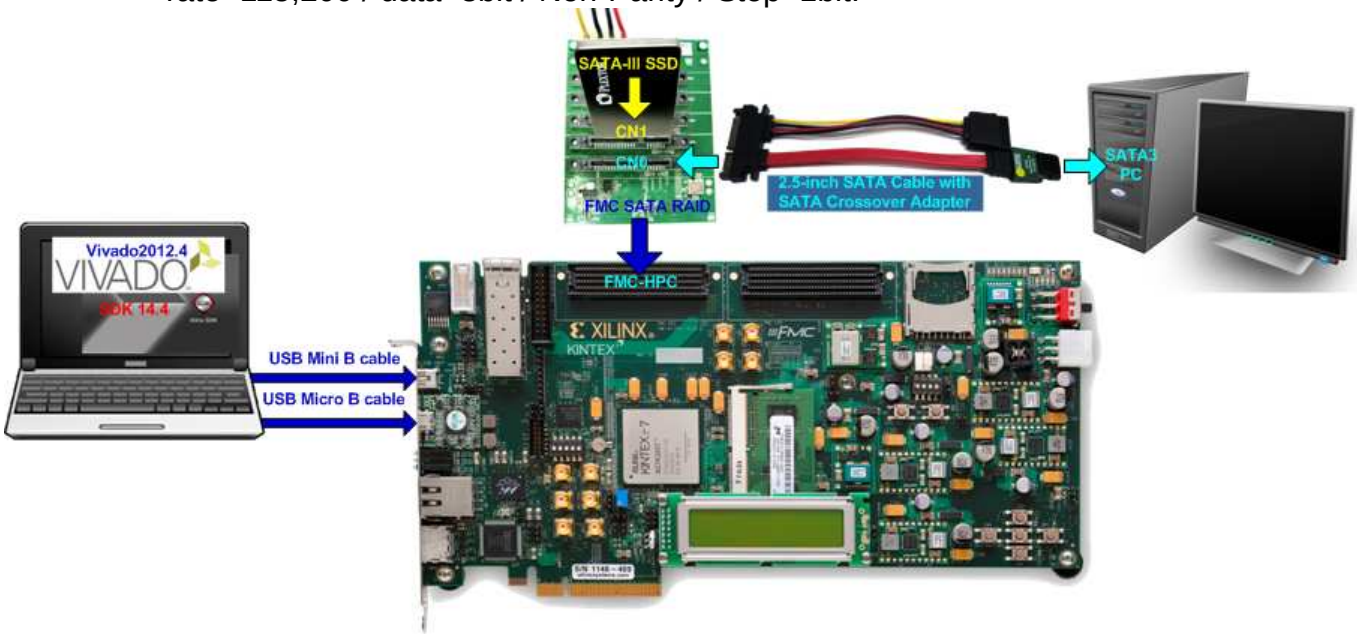

Figure 2 Reference design environment

Refer to “SATA-IP Bridge Demo Instruction on KC705” for operation procedure of this reference design.

3. Hardware description

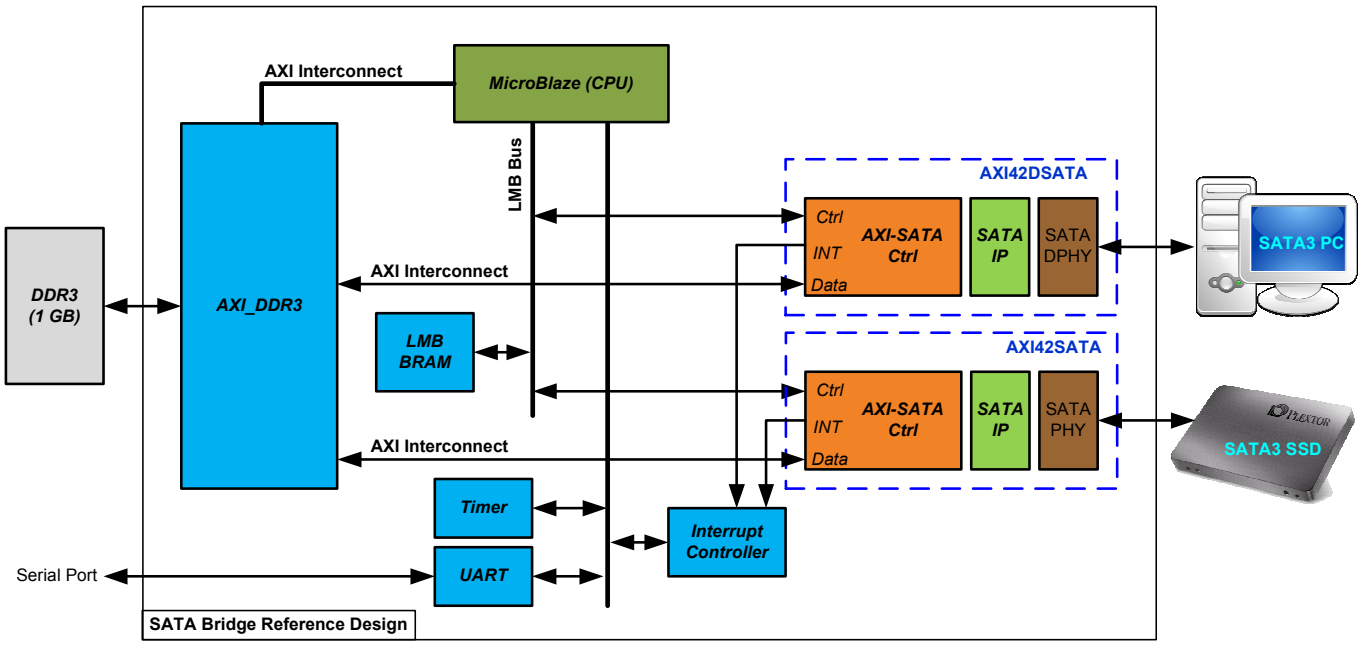

Figure 3 Block diagram of the reference design

- SATA IP Bridge design implementation on Kintex7 FPGA

To operate SATA Bridge function, two SATA-IPs are connected to EDK system to operate SATA Host and SATA Device. Different PHY module is designed to support Host or Device mode while SATA-IP and AXI-SATA Ctrl module are similar. SATA packet sent from SATA3 PC/SATA3 SSD will be stored to DDR3. MicroBlaze will decode SATA packet and then send control signals to forward the packet to another side which is SATA3 SSD or SATA3 PC.

The system design to operate SATA protocol can be split into several modules following the protocol layer, as shown in Figure 4.

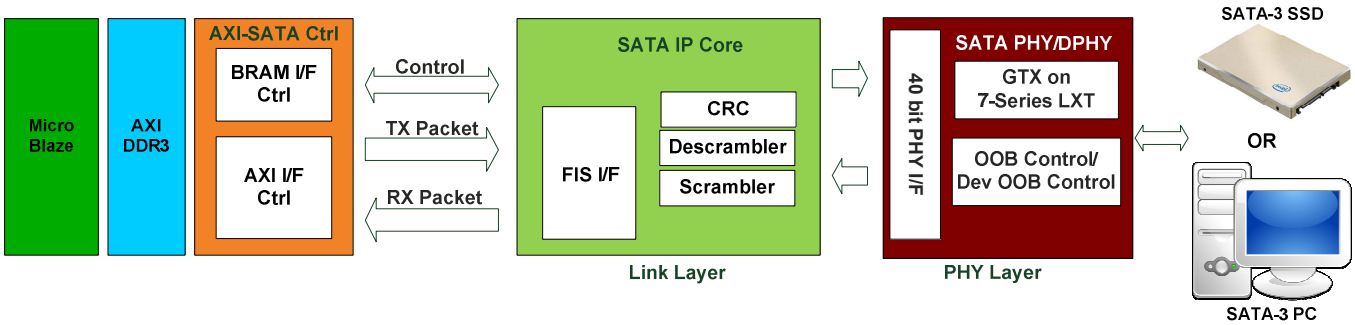

Figure 4 AXI42SATA Block Diagram

- PHY Layer

This layer is designed by using GTX module (built-in high speed serial circuit) of Kintex-7 device, operating with logic control to generate OOB sequence and initialization sequence of physical layer. There are two different PHY designs to operate Host and Device mode.

For host mode, state machine to control OOB is designed in "oob_control.vhd" which is sub-module of "sata2phy_k7.vhd", the top layer of PHY source code which also includes Xilinx transceiver module and the interface logic.

For device mode, OOB sequence in "dev_oob_control.vhd" will be reversed to host mode operation. Similar to host mode, the top module, "sata2dphy_k7.vhd", includes OOB module and transceiver parameter setting.

PHY design in 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. The host and device operation in this module are similar. More details about SATA-IP interface are described in "dg_sata_ip_datasheet_kt7_en" document.

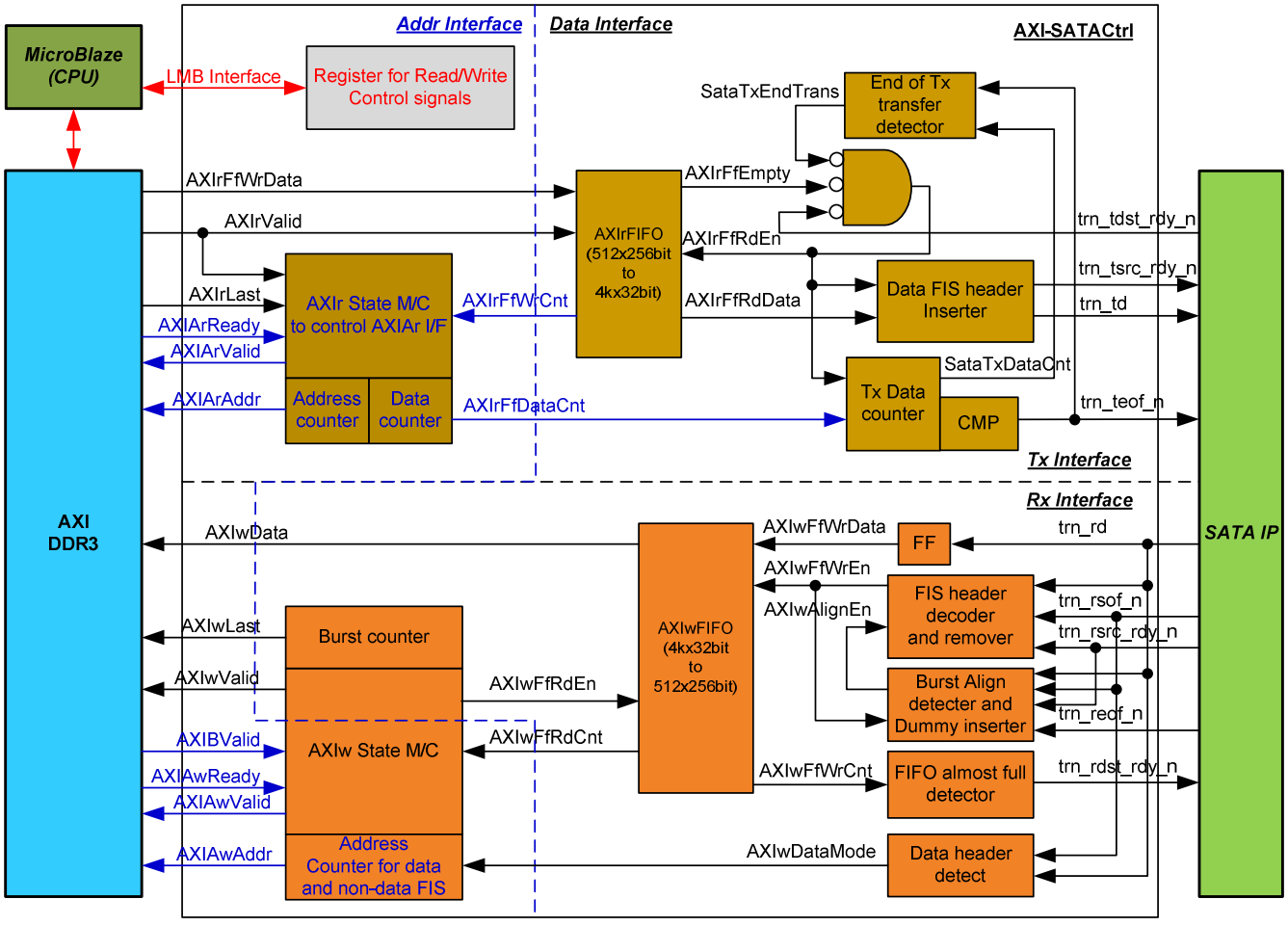

Figure 5 Block diagram of AXI-SATACtrl

- Transport Layer by AXI-SATACtrl

For this layer, the host and device mode use the same design structure, but connecting to different PHY module. The details of the module are follows.

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.

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. for sector alignment.
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. for sector alignment.
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 for Host is 0x0001_0000, Base Address for Device is 0x0002_0000)

Register map of control signals to interface with MicroBlaze by LMB bus is shown in Table 1. Main operations of these registers are to define DDR address for transferring FIS data, transfer length, and FIS type (data or non-data). The status of operation can be monitored by MicroBlaze accessing these registers.

For data transfer with DDR3, it uses AXI4 bus interface. AXI4 bus can be divided into four groups, i.e. AXIAr for read command request, AXIr for read data transferring, AXIAw for write command request, and AXIw for write data transferring. The requests for both read and write command are controlled by state machine. Data bus size of AXI4 is 256-bit, so the FIFO must be used for data bus converting between 256-bit and 32-bit (SATA-IP bus size). The sequence of read and write operation are follows.

For read operation from AXI to SATA, the operation is started by MicroBlaze. After FIS or data is prepared in DDR3, MicroBlaze sets CONTROL register to start data or non-data FIS transferring to SATA with setting the packet size. If data FIS is sent, Data FIS header will be auto-added by internal logic and then followed by the data from DDR3. AXIrFIFO is applied to convert data bus size and for data flow control. If FIFO is almost full, state machine will pause to request next data from DDR3 and wait FIFO space available enough. Also, the logic at SATA side will monitor empty flag of FIFO to read and send data out to SATA-IP.

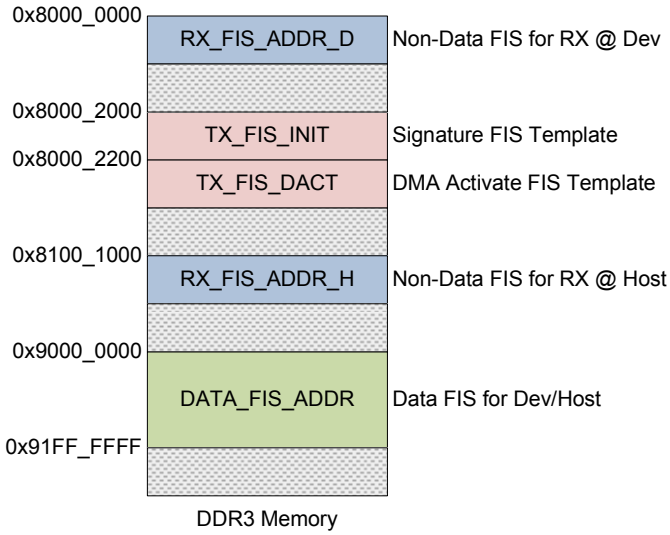

Figure 6 DDR3 Memory map in the demo

For write operation from SATA to AXI, the operation is started by SATA. DDR3 address to store FIS from SATA is pre-defined by MicroBlaze. Two different DDR3 areas are defined for storing Data FIS and non-DATA FIS packet, as shown in Figure 6. Data FIS is stored to DATA_FIS_ADDR area while non-DATA FIS is stored to RX_FIS_ADDR_D/H following the operation mode. AXIwFIFO is used for data flow control and data bus size converting. If FIFO is almost full, SATA packet from SATA-IP will be paused transferred. If total data in SATA packet from SATA-IP is too small size and not aligned word (less than 256-bit), the dummy word will be filled to FIFO by internal logic. At the read side, state machine will be always monitored FIFO count to check that the data is much enough, and then send request and forward data out to DDR3 through AXI4 bus. The FIS header of every SATA packet is decoded to check FIS type that is non-data or data format, and then select the correct DDR3 address to store the SATA FIS.

Data transactions in both directions are size-fixed to 256-bitx16 beat (512 byte) for simple design logic and high performance transfer. AXI-SATACtrl logic operating with SATA-IP and PHY layer code are stored in “AXI42SATA.vhd” for host mode and “AXI42DSATA.vhd” for device mode, provided to user as delivery item.

Note: TX_FIS_INIT and TX_FIS_DACT area in Figure 6 are designed to be SATA FIS template to improve transfer performance for Bridge design. More details are described in the next topic.

4. Software description

- SATA Bridge operation

SATA Bridge firmware is modified from SATA Device and Host demo reference design. Unlike both demo designs, SATA FIS (Frame Information Structure) is typically not created by MicroBlaze, but MicroBlaze is only forwarded SATA FIS from Host to Device or Device to Host. But the FIS still need to be decoded by MicroBlaze before forwarding for analyzing the command sequence and control data flow to the main memory by DMA controller within both side designs.

MicroBlaze in the SATA Bridge will operate as below sequence.
(1) Receive FIS Data (command) from PC
(2) Decode and execute the ATA command
(3) Forward FIS Data (command) to HDD/SSD
(4) Forward additional FIS data if necessary (for data command)
(5) Wait FIS Data (status) from HDD/SSD
(6) Forward FIS Data (status) to PC

- Software of reference design

Software source code of Bridge reference design is stored in “sata_bridge.c”. For simple design, the demo does not support some features, i.e.

- TRIM
- NCQ (Native Command Queuing)
- More than 1 sector PIO Read and Write command

Typically, MicroBlaze will only forward FIS Data from one side to another side except READ DMA (EXT) and WRITE DMA (EXT) command which needs to optimize data sequence to achieve high performance.

- READ DMA (EXT)

The brief sequence of bridge firmware is follows.
(1) Decode total data size from Command FIS
(2) Forward Command FIS from the device mode (connecting to PC) to the host mode (connecting to HDD/SSD)
(3) Wait data from HDD/SSD
(4) If data is much enough, MicroBlaze will set register at device side (PC) to send FIS Data from DDR3.
(5) Step (3) – (4) are repeated in the loop until end of data transfer.
(6) Wait FIS Data (status) from HDD/SSD
(7) Forward FIS Data (status) to PC

- WRITE DMA (EXT)

The brief sequence of bridge firmware is follows.
(1) Decode total data size from Command FIS
(2) Create and Send DMA Activate FIS to the device mode (PC)
(3) Wait data from PC
(4) Forward data to the host side (HDD/SSD) when data is much enough and DMA Activate FIS is received from HDD/SSD.
(5) Step (3) – (4) are repeated in the loop until end of data transfer
(6) Wait FIS Data (status) from HDD/SSD
(7) Forward FIS Data (status) to PC

- Performance result

Figure 7 shows the example performance when running disk benchmark through SATA Bridge reference design.

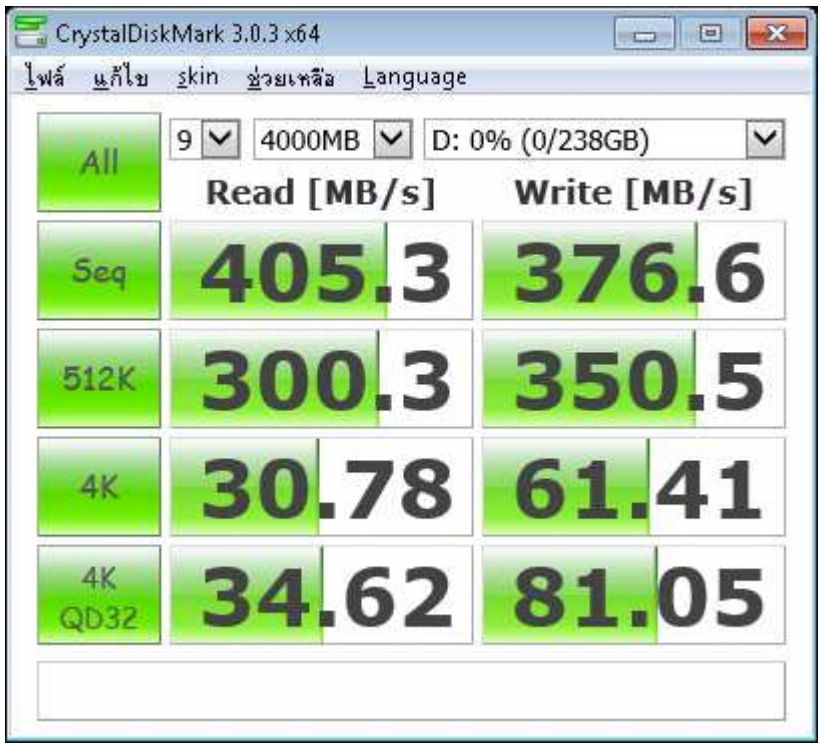

Figure 7 Operation result sample screen

5. Revision History

Revision	Date	Description
1.0	03-Mar-14	Initial release