8-ch RAID0 Design by using SATA Host IP Manual

Rev1.0 9-Jun-15

1 Overview

RAID0 system uses multiple storages to extend total storage capacity and increase write/read performance to be N times. Assumed that total number of disk is N, during write direction data from the host will be split into N stripes, so only one stripe will be stored to one disk, as shown in Figure 1. Since minimum transfer size of one SATA device is one sector, the minimum stripe size is equal to 512 byte.

This demo uses 8 SATA devices for RAID0 system and stripe size for one disk is equal to one sector. So, total disk capacity will be equal to 8 times of one device and the performance will be almost 8 times. Comparing to RAID0 demo implemented by CPU firmware, the overhead time to process each data packet by SATA Host-IP is less, so total write performance by using SATA Host-IP will be better than using CPU processing.

Note: Write command consists of many packets transferring between the host and the device while read command consists of less packets. So, the overhead from packet processing by CPU will be much effect in Write direction.

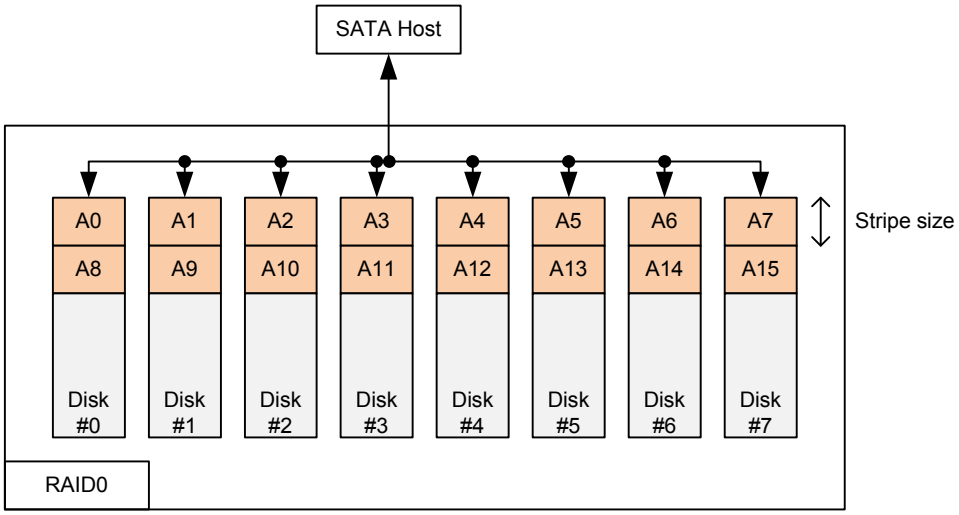

Figure 1 RAID0 Data Format

As shown in Figure 2, DDR3 are used to be big data buffer to transfer very high speed data in sustain rate. Too small data buffer size will reduce the performance. In the demo, there are two masters interfacing with DDR3, i.e. User Module and RAID0 system. User Module is the test logic to generate and verify test data at specified rate. In the demo, user can select test pattern, test size, and test speed of User Module through DIPSW input. 512-bit FIFOs are used to be small data buffer before transferring with DDR3. AvMasterCtrl are used to be interface module between User Module/RAID0 system and DDR3.

For RAID0 system, SATAFIFOs are additional required to convert data bus size between 512-bit (DDR3 bus size) and 32-bit (SATA bus size). Also, SATAFIFOs are used to be data buffer for RAID0 controller to split and combine data of each SATA channel. For write direction, every 8 sectors in RAIDTxFIFO are split into 8 parts and one SATATxFIFO will be stored only one sector. For read direction, one sector from 8 SATARxFIFOs are read and fed to RAIDRxFIFO in RAID0 format. More details about each module are described in the next topic.

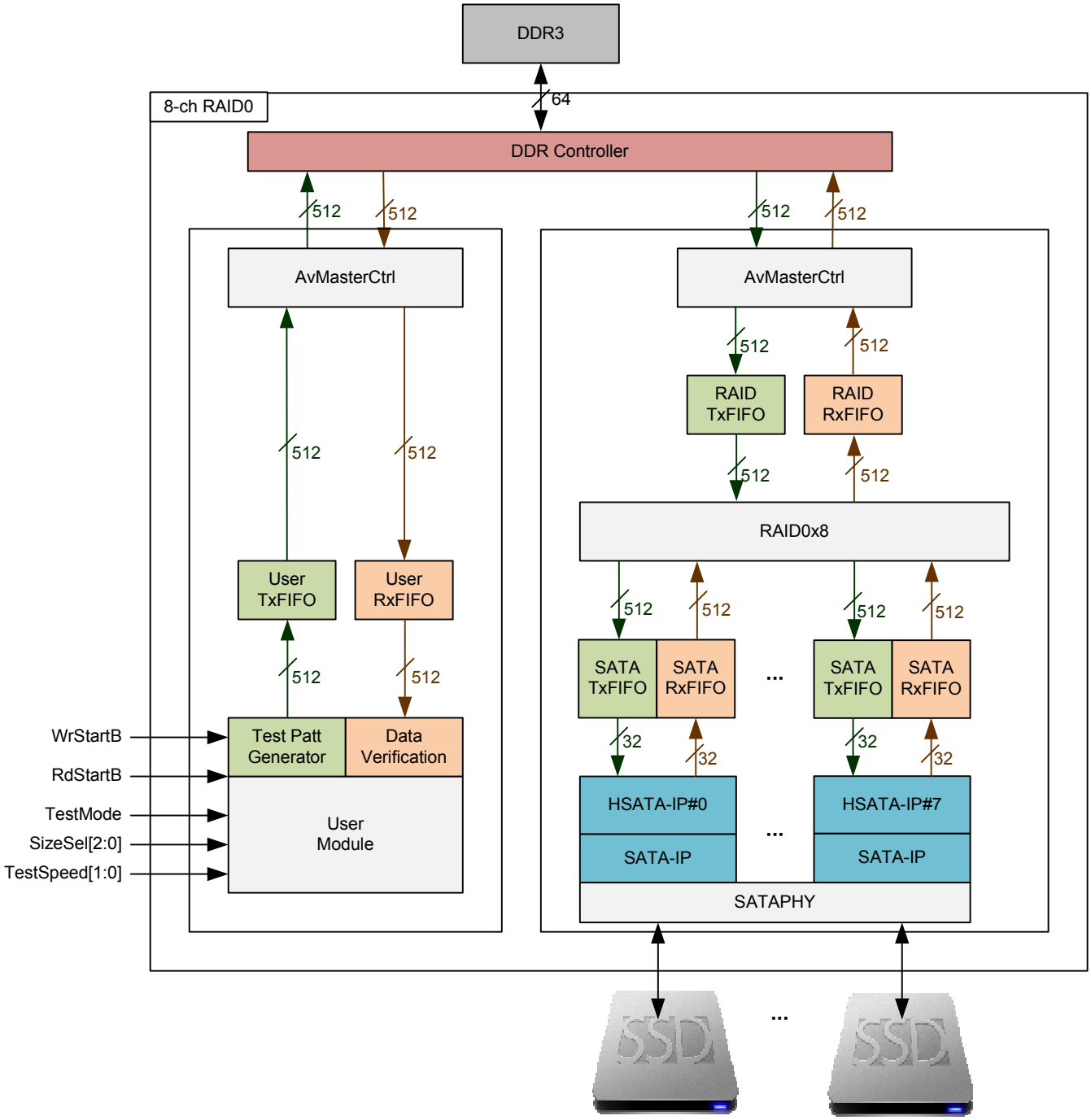

Figure 2 8-ch RAID0 Demo System by using SATA Host IP

2 RAID0x8

As shown in Table 1, User interface of RAID0 controller is almost similar to SATA Host IP. But one-clock pulse is used to be command start signal instead of using command request like SATA Host IP. Total transfer size in one command is extended to 40-bit, so maximum transfer size in one command is extended to 512 TB. Maximum transfer size of one SATA Host IP is 32 MB, so state machine in RAID0 controller will generate many command requests to SATA Host IP when data transfer size is larger than 8x32MB. The error status of each SATA channel is direct mapped to be the output of RAID0 controller.

FIFO status input to RAID0 controller is one-bit ready flag. For FIFO read, the ready flag must be asserted to '1' when at least 1 sector data is available in FIFO. For FIFO write, the ready flag must be asserted to '1' when space area in FIFO is more than 1 sector.

SATA Host IP signal interface has 8 sets of SATA Host IP and SATA FIFO connection. Only CH#0 is displayed to be example in Table 1.

2.1 Port Description

Table 1 Signal Description of Raid0x4

Signal	Dir	Description
User Interface		
RstB	In	Reset signal. Active low. Please use same reset signal as SATA Host IP.
Clk	In	User clock. Must use the same clock as SATA Host IP.
RaidCmd	In	Command. '0': Write data to RAID0, '1': Read data from RAID0
RaidCmdStart	In	1-clock period pulse to request the new command. Can be asserted only when RAID0 is Idle (RaidBusy='0'). Assert with valid value on RaidCmd/RaidAddr/RaidLength signals.
RaidAddr[47:0]	In	RAID address to write/read in sector unit (512 byte).
RaidLength[39:0]	In	Total transfer size in sector unit (512 byte).
RaidErrorClr	In	Direct connection to generate all bits of SataErrorClr output signal.
RaidBusy	Out	RAID0 busy status. New request is not allowed if this signal is asserted to '1'.
RaidLBASize[47:0]	Out	Total RAID0 capacity in sector unit (512 byte). Calculated by 8 times of LBASize0 input signal.
RaidError	Out	Assert when RaidErrorNo signal is not equal to 0.
RaidErrorNo[63:0]	Out	Bit[5:0] – Direct connection from SataErrorNo at CH#0 Bit[13:8] – Direct connection from SataErrorNo at CH#1 Bit[21:16] – Direct connection from SataErrorNo at CH#2 Bit[29:24] – Direct connection from SataErrorNo at CH#3 Bit[37:32] – Direct connection from SataErrorNo at CH#4 Bit[45:40] – Direct connection from SataErrorNo at CH#5 Bit[53:48] – Direct connection from SataErrorNo at CH#6 Bit[61:56] – Direct connection from SataErrorNo at CH#7 Other bits are unused and always set to 0.

Signal	Dir	Description
RAID FIFO Interface		
RaidTxFfRdRdy	In	Assert when at least 1 sector data is available in Raid transmit FIFO. Created by using upper bit of Raid transmit FIFO read counter.
RaidTxFfRdEn	Out	Read valid of Raid transmit FIFO
RaidTxFfRdData[511:0]	In	Read data returned from Raid transmit FIFO. Valid after RaidTxFfRdEn asserted.
RaidRxFfWrRdy	In	Assert when space area in Raid received FIFO is more than 1 sector. Created by using upper bit of Raid received FIFO write counter.
RaidRxFfWrEn	Out	Write data valid of Raid received FIFO
RaidRxFfWrData[511:0]	Out	Write data bus of Raid received FIFO. Synchronous to RaidRxFfWrEn.
Host SATA IP Interface (Show only CH#0 to be example)		
SataCmd[0]	Out	The description is described in SATA Host IP datasheet
SataCmdReq[0]	Out	
SataAddr[0][47:0]	Out	
SataLength[0][16:0]	Out	
SataBusy[0]	In	
LBASize0[47:0]	In	
SataError[0]	In	
SataErrorNo[0][5:0]	In	
SataErrorClr[0]	In	
SATA FIFO Interface (Show only CH#0 to be example)		
TxFfRdy[0]	In	Assert when space area in SATA transmit FIFO is more than 1 sector. Created by using upper bit of SATA transmit FIFO write counter.
TxFfWrEn[0]	Out	Write data valid of SATA transmit FIFO
TxFfWrData[0][511:0]	Out	Write data bus of SATA transmit FIFO. Synchronous to TxFfWrEn[0].
RxFfRdy[0]	In	Assert when at least 1 sector data is available in SATA received FIFO. Created by using upper bit of SATA received FIFO read counter.
RxFfRdEn[0]	Out	Read valid of SATA received FIFO
RxFfRdData[0][511:0]	In	Read data returned from SATA received FIFO. Vaild after RxFfRdEn asserted.

2.2 Timing Diagram

After reset is released, RAID0 will wait until all SATA Host IPs initialize SATA devices completely by monitoring SATABusy signal, as shown in Figure 3. RaidBusy is de-asserted to '0' when all SATABusy=0. Raid capacity is valid at the same clock that LBASize0 valid because it can be simply calculated by multiply by 8. After that, RAID0 controller is ready to receive command from user.

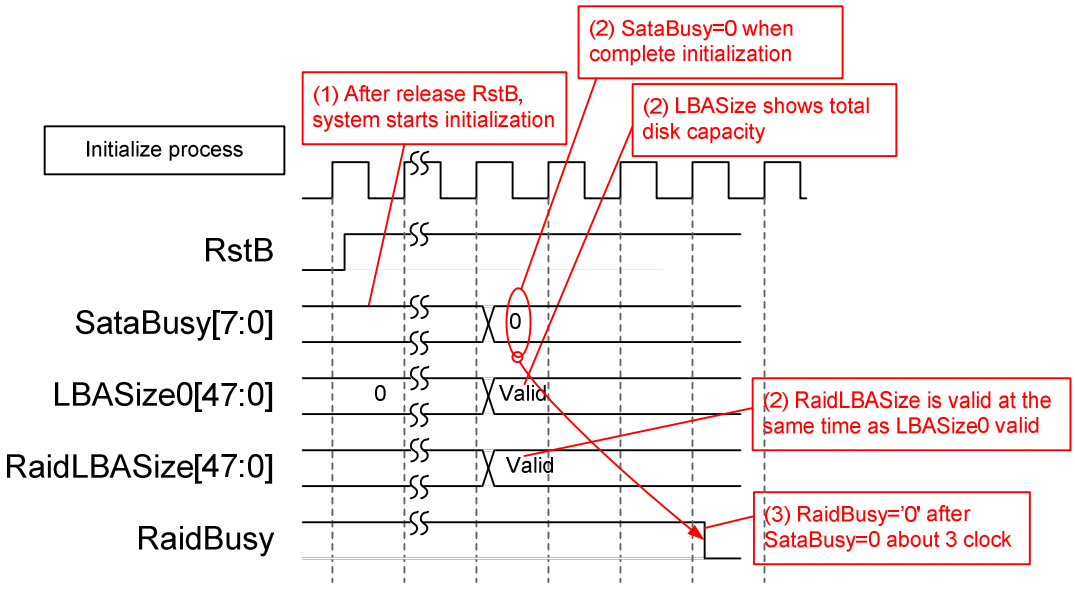

Figure 3 RAID0 Initialization Timing Diagram

For user interface, RaidCmdStart pulse for new request is asserted with the valid value of RaidCmd, RaidAddr, and RaidLength, as shown in Figure 4. In the next clock, RaidBusy will be asserted from '0' to '1' and hold this value until complete current command.

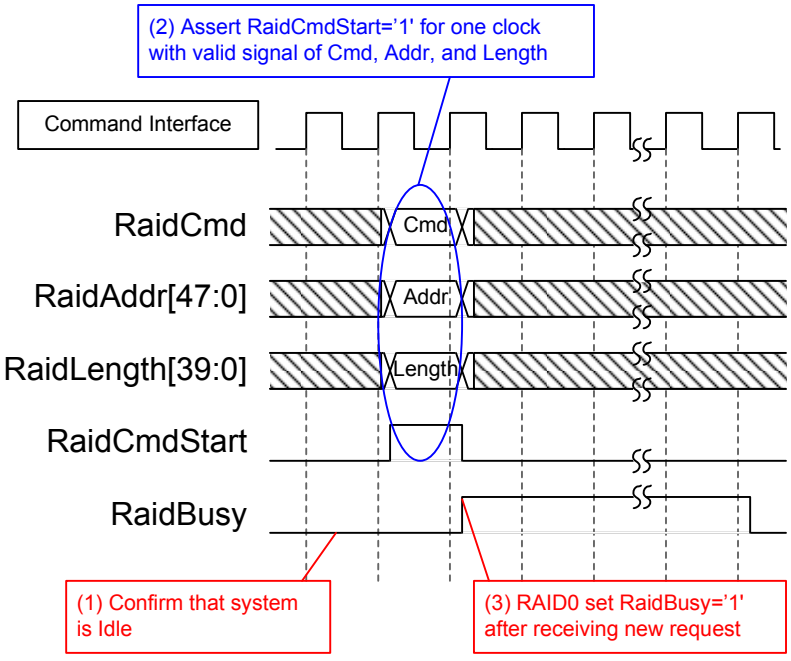

Figure 4 RAID0 Command Interface Timing Diagram

When user sends write command to RAID0 controller, data will be forwarded from RaidTxFf to TxFf[0]-[7] sector by sector, as shown in Figure 5. Before data forwarding, ready flag status of both FIFOs will be monitored to confirm that at least 1 sector data is available in RaidTxFf and more than 1 sector space area is TxFf. One sector data is transferred by using 8-beat of 512-bit burst size. When end of each burst, the active channel will be switch to next channel. In this example, next transfer will use TxFfRdy[1] and TxFfWrEn[1] for interfacing with CH#1 which is the next channel of CH#0.

The first active channel for RAID0 operation depends on 3-lower bits of RaidAddr bus signal. These 3-lower bits can be referred to start active channel directly, i.e. “000”=CH#0, “001”=CH#1, “010”=CH#2, …, “111”=CH#7. Data will be burst transfer sector by sector until total transfer size is equal to RaidLength input from user.

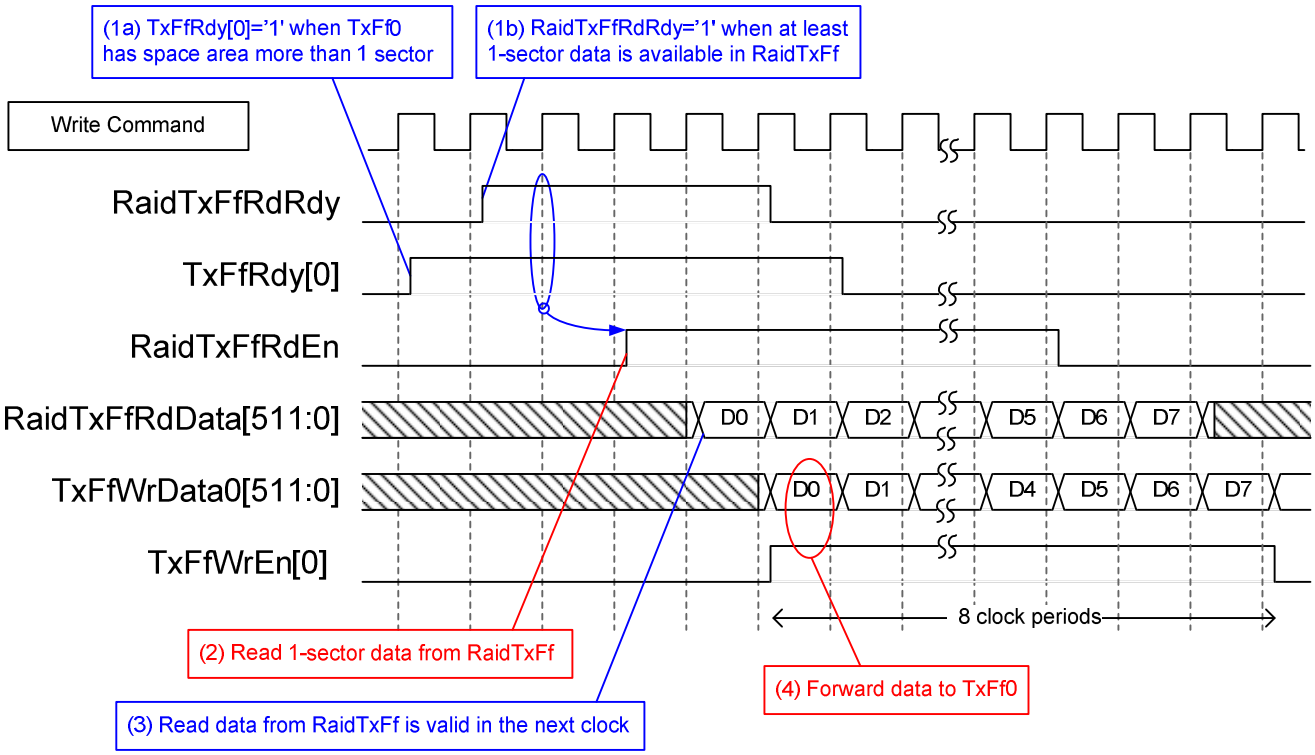

Figure 5 FIFO Timing Diagram of Write Command when active SATA is CH#0

When user sends read command to RAID0 controller, data will be forwarded from RxFf[0]-[7] to RaidRxFf, as shown in Figure 6. Data will be forwarded sector by sector, similar to write command. When both FIFOs are ready, RaidRxFfWrEn[0] will be asserted for 8-clock period. This signal is used to generate RxFfRdEn[0] which is active channel in the next clock, and used to generate real write enable of RaidRxFf in the next three clocks. Write data to RaidRxFf is received from RxFifo data output of active channel, selected by data multiplexer.

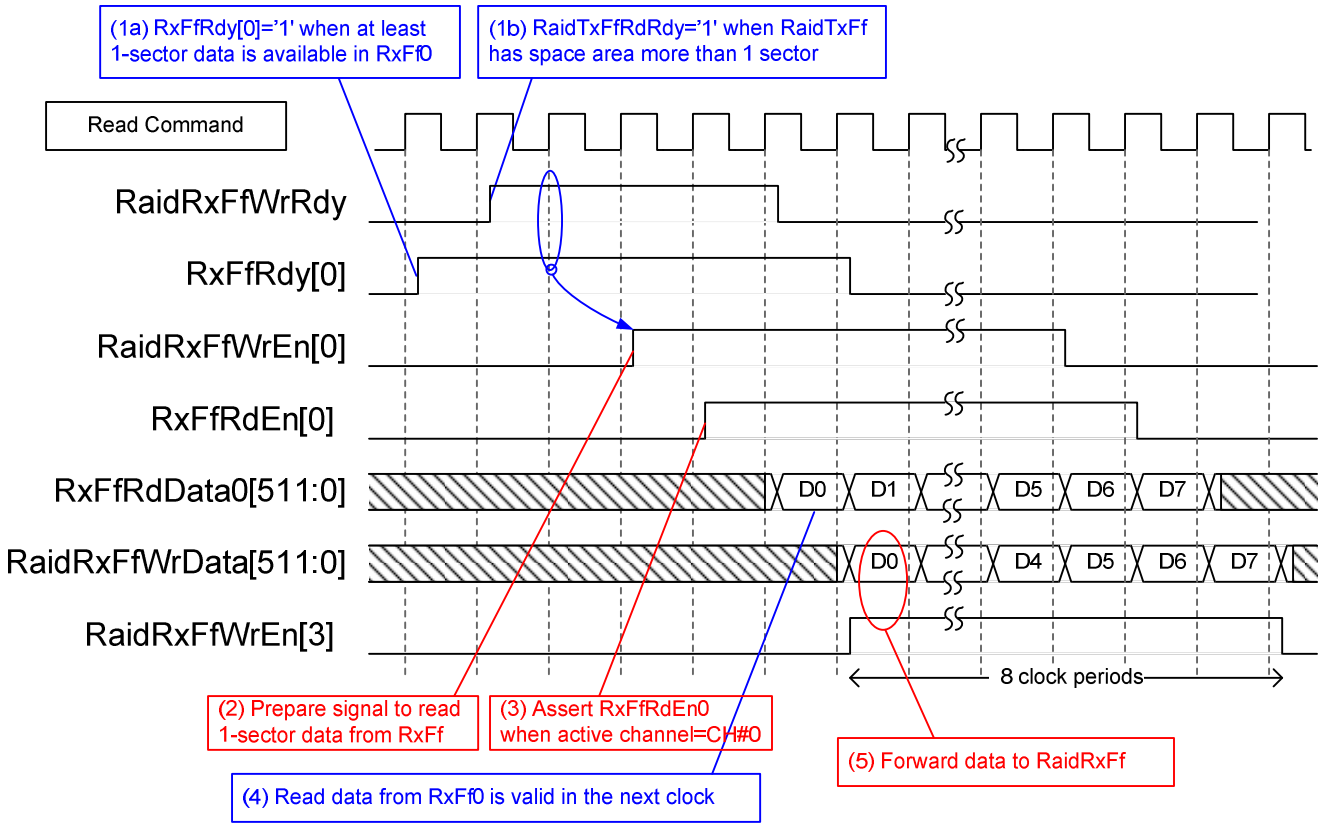

Figure 6 FIFO Timing Diagram of Read Command when active SATA is CH#0

2.3 State Machine

The main controller within RAID0 module is designed by using State machine which has the sequence as shown in Figure 7. Reset state is designed to wait until all SATA Host IPs initialize SATA device completely by monitoring SATABusy signal. Next, it will wait request from user to start data forwarding between RaidFifo and TxFifo/RxFifo.

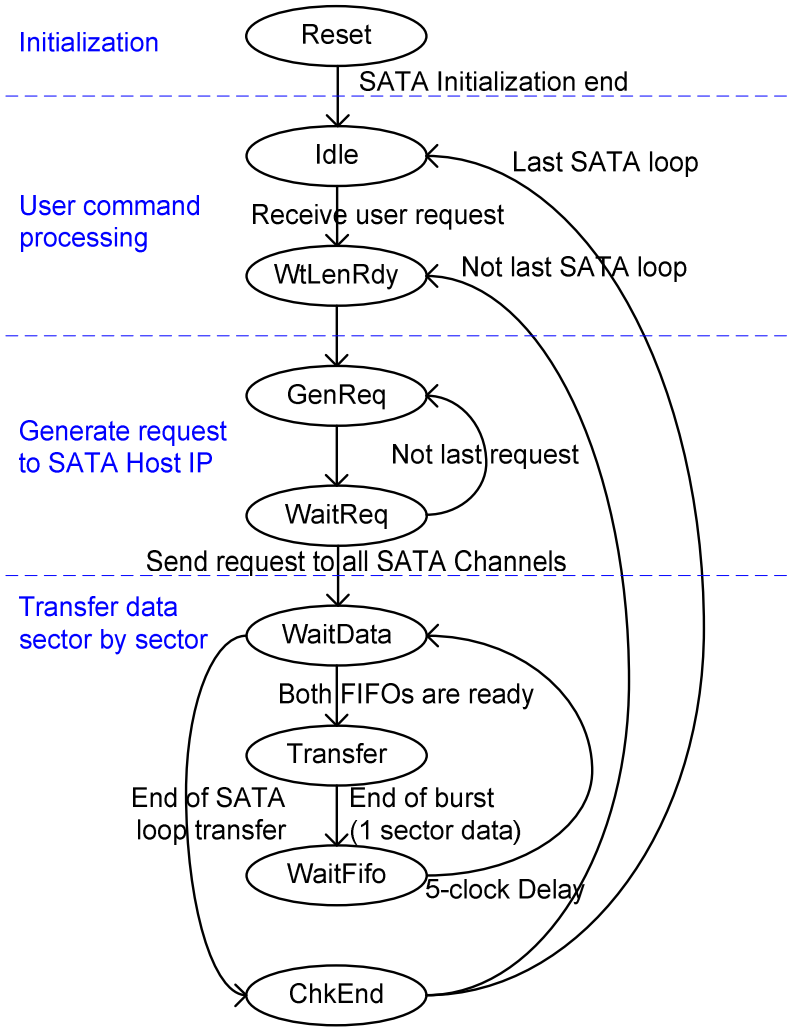

Figure 7 RAID0 controller State Machine

Maximum transfer size of one SATA Host IP is equal to 32 MB, so the maximum request size in GenReq state for 8 channels is equal to 8x32 MB = 256 MB size. To support 40-bit transfer length, state machine inside RAID0 controller will repeat to send SATA Host IP request until total SATA request size equal to RaidLengh. WtLenRdy state is designed to prepare address and length of SATA Host IP which will be valid at the same clock with asserting the request in GenReq state. The address and length of each channel may be different when user input or length is not aligned 8-sector. So, WaitReq state is designed to re-calculate address and length of the next channel. When RaidLength>= 8, GenReq/WaitReq will run in loop for eight times to generate request to all SATA channels. But if RaidLength is less than 8, only some SATA Host IPs will be received the request.

After sending request completely, it will change to data transfer step. WaitData is designed to check ready status of RaidFifo and TxFifo/RxFifo depending on data direction. Transfer state will generate read enable of FIFO for 8-clock periods to forward one sector data from source FIFO to destination FIFO. WaitFifo state is delay state to hold the operation about 5 clock periods. This delay is required to support data latency within data forwarding process and change the active channel selector for next sector transfer.

State will change from WaitData to ChkEnd when total data transfer size is equal to 8x32 MB or less (for last loop which is not aligned to 8x32 MB). In ChkEnd state, state will go back to Idle state if total remain RAID transfer size is equal to 0. If not equal to 0, it will go to WtLenRdy to generate next request to SATA Host IP.

3 AvMasterCtrl

This module is designed to transfer 512-bit data from FIFO interface to Avalon bus master interface. The module supports three burst sizes for both write and read direction, i.e. 8x512-bit, 16x512-bit, and 32x512-bit. Bigger burst size will reduce the overhead time of each burst and increase transfer performance. State machine in this module is shown in Figure 8.

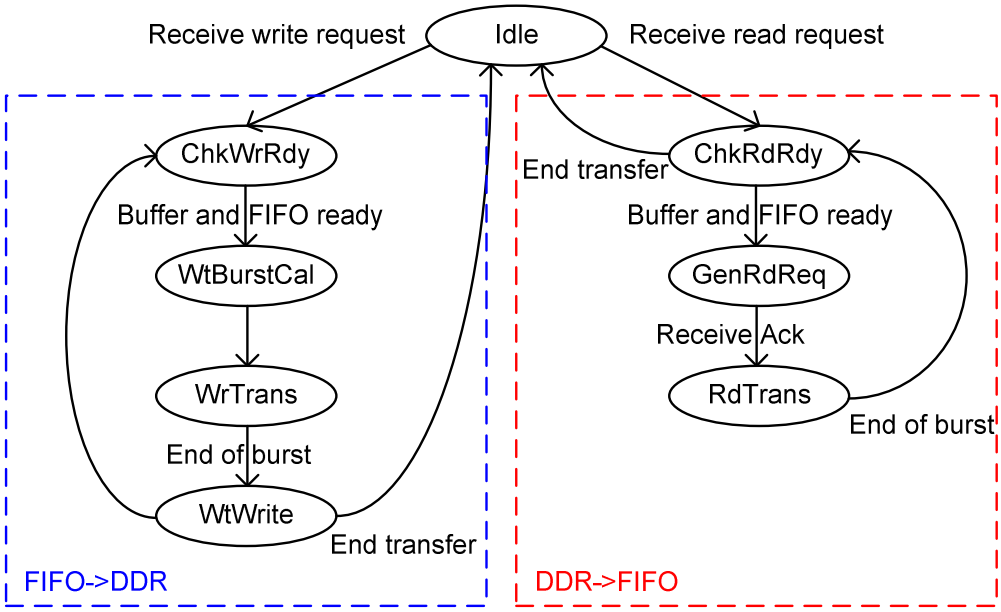

Figure 8 AvMasterCtrl State Machine

State machine is split into two operations, i.e. write DDR on the left side, and read DDR on the right side. When user sends the request to write DDR, state machine will go to ChkWrRdy state. State will change to WtBurstCal state if at least 1 sector data available in FIFO and more than 1 sector space available in DDR. WtBurstCal state is designed to add latency in state machine to calculate the biggest burst size for current transaction, depending on transfer size, data available size in FIFO, and space area in DDR. During WrTrans state, data will be transferred from FIFO to DDR until end-of-burst. WtWrite is used to check remain transfer size. If all data are transferred, it will go back to Idle state for waiting new command. If not, next data burst transfer will be started.

For read request from DDR to FIFO, state will go to ChkRdRdy state. Similar to write direction, it will go to next state after at least 1 sector data available in DDR and more than 1 sector space available in FIFO. But if total data are transferred completely, it will go back to Idle state. Burst count value will be calculated from remain transfer size, data available in DDR, and space available in FIFO within ChkRdRdy state. GenRdReq is used to generate the request to DDR and wait command acknowledge. During RdTrans state, data will be transferred from DDR to FIFO until end-of-burst.

4 UserLogic

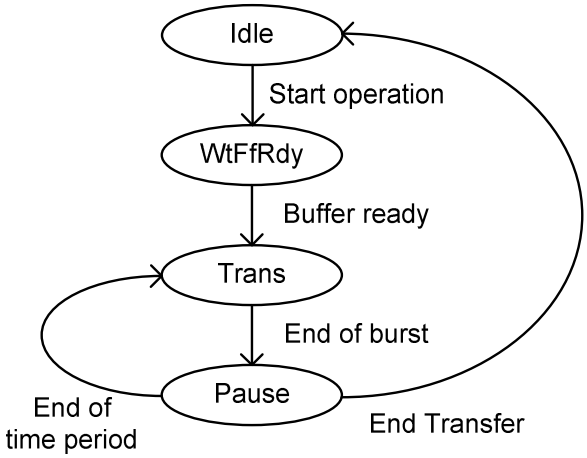

Figure 9 UserLogic State machine

In the demo, user can select data speed by using 2-bit DIPSW input. Data speed is adjusted by adjusting delay time in Pause state after one burst transfer. Four speed is designed, i.e. 2400 MB/s, 2800 MB/s, 3200 MB/s, and 3600 MB/s. Increase delay time in Pause state will reduce data throughput of User Module. Trans state is designed to generate write or read enable to FIFO for 8 clock period or one sector size. WtFfRdy state is designed to find the proper start point after receiving user command. In the demo, it will check that FIFO has space area much enough for write test, while it will check that data in DDR is much enough for read test.

5 Revision History

Revision	Date	Description
1.0	9-Jun-15	Initial version release