Tests for Data Storage and Manipulation (DSM) boards for STAR Trigger Level 0

1. Requirement: Test DSM Look-Up Table (LUT) for 16 8 bit words.

   Justification: This test is performed to ensure that the LUT table is being correctly loaded and LUT corrections are correctly being applied to data and data is correctly being addressed in the Data Storage area.

   Implementation: Load DSM 16 8 bit Look-Up Table (LUT). Input stream of zero's with 16 channels of data interspersed. Perform LUT correction. Read data from DSM Input Buffer (the Data Storage).

2. Requirement: Test DSM Look-Up Table (LUT) for 24 5 bit words. Note this will be implemented based on the decision concerning the MWC detector methodology.

   Justification: This test is performed to ensure that the LUT table is being correctly loaded and LUT corrections are correctly being applied to MWC wire data.

   Implementation: Load DSM 8 16 bit Look-Up Table (LUT). Input stream of zero's with 24 channels of 5 bit/channel data interspersed. Perform LUT correction. Read data from DSM input buffer.

3. Requirement: Test DSM L1/L2 shift register.

   Justification: This ensures that shift of data from the main storage on the DSM board to the L1/L2 registers is being performed correctly.

   Implementation: With loaded 16 8 bit LUT. Input zero stream plus 16 channels of data. Perform LUT correction. Shift data from DSM Storage to L1/L2 Register. Read data from register.

4. Requirement: Check that FIFO control is performed properly.

   Justification: Ensure that FIFO is working properly so that it can be used as DSM output.

   Implementation: Clear the FIFO and check that the FIFO_Status bits and LED's show that it is empty. Write 4096 blanks to the FIFO. Check that the FIFO_Status bits and LED's show that it is full. Read out blanks and check that the correct number are read out before the FIFO is empty. Repeat using an intermediate number of blanks, and check that the status bits show that it is neither full nor empty. Repeat the entire cycle over many hours to check for robustness and rare errors.

5. Requirement: Test addition algorithm on DSM.

Justification: Ensure correct answer for addition algorithm from FPGA engine.

Implementation: Input zero stream + 16 channels of 8 bit information. Perform LUT correction. Apply addition algorithm. Output 16 bit sum of input data. Read sum at DSM FIFO.

6. Requirement: Test DSM division algorithm.

Justification: Ensure correct answer for division algorithm from FPGA engine.

Implementation: Input zero stream + 16 channels of 8 bit information. Perform LUT correction. Apply division algorithm and output 16 bit result of input data. Read result at DSM FIFO.

7. Requirement: Test minimum finding algorithm on DSM.

Justification: Ensure correct answer for algorithm to determine minimum from FPGA engine.

Implementation: Input zero stream + 16 channels of 8 bit information. Perform LUT correction. Apply algorithm to determine minimum and output 16 bit result of input data. Read result at DSM FIFO.

8. Requirement: Test LUT correction on complete data stream.

Justification: This test is performed to ensure that the data is correctly being addressed in the Data Storage area and can be accessed using given DSM memory pointers. Also that LUT has been loaded correctly.

Implementation: Load DSM 16 8 bit Look-Up Table (LUT). Input stream of 16 channels with input numbers 0-255. Perform LUT correction. Read data from DSM Input Buffer.

9. Requirement: Test L1/L2 shift register on data stream.

Justification: This ensures that shift of data from the main storage on the DSM board to the L1/L2 registers is being performed correctly, thus pointers are correct. This guarantees that the L1 and L2 trigger subsystems will access the correct information for the given event.

Implementation: With loaded 16 8 bit LUT. Input 16 channels of streamed data. Perform LUT correction. Shift data from DSM Storage to L1/L2 Register. Read data from L1/L2 register.

10. Requirement: Test addition, division and minimum finding algorithms as mentioned above on data stream.

Justification: Ensure correct answer for addition, division and minimum algorithms from FPGA engine within some given time frame. Timing tests to ensure algorithm can be performed within time window (canonically 80 ns).

Implementation: Input 16 channels of 8 bit stream data. Perform LUT correction. Apply addition, division and minimum algorithm to output 16 bit sum of input data. Read sum at DSM FIFO. Perform timing tests.

11. Requirement: Apply above mentioned tests on 24 5 bit input stream. (Note: This is dependent on MWC decision.)

Justification: Ensure MWC sums/divisions can be performed in allotted time.

Implementation: Input 24 channels of 5 bit stream data. Perform LUT correction. Apply addition, division and minimum algorithm to output 16 bit sum of input data. Read sum at DSM FIFO. Perform timing tests.

12. Requirement: Test FIFO as a “space holder”.

Justification: Guarantee FIFO is performing “space holder” functionality correctly.

Implementation: Load FIFO with a number of blanks (spaces). Input 16 channels of 8 bit stream data. Perform LUT correction. Apply addition algorithm to output 16 bit sum. Read FIFO output and check that the correct number of blanks are read out before the data.

13. Requirement: Test “real data” input on boards. Above tests were performed with input from the simulation memory.

Justification: Ensure correct load/read functions of real data.

Implementation: Perform a collection of previous tests utilizing data from the “real data” storage on DSM board.

14. Requirement: Check that writing/reading all the DSM memories (Simulation Memory, LUT's, Input Buffer, FPGA Memory (configuration file)) is robust and NEVER fails.

Justification: Ensure that writing to memories is correct.

Implementation: Write random numbers to a block of memory and read them back. Produce error message if what is read back is not the same value as that written. The address should be chosen randomly within the block (or at least not be sequential) so that both address and data change for every read/write cycle. Repeat for many hours to thoroughly test all blocks.

15. Requirement: Check that resetting and incrementing address pointer is correct.

Justification: It is necessary to reset the address pointers so they can all be aligned at the same point. Need to be able to increment pointers on some DSM's in the final tree so that all the data from one RHIC strobe is at the same address.

Implementation: Reset the address pointer to zero, latch it to a register, read the register and check that the address is zero. Increment the address by one, latch and read and verify that it is now 1. Repeat this many times, checking at each stage. Repeat the entire cycle over many hours to ensure robustness and check for rare errors.