DSM Testing Plan
E.G. Judd
July 1, 2021

This plan assumes all DSM2 boards have been tested in stand-alone mode, including Out-In loopback tests for both the parallel TTL I/O and the serial fiber I/O, and are known to be working.

It covers tests that should be done at STAR during Run21 as well as remaining firmware development and the tests that can be done during the shutdown before Run22. Firmware development will be happening continuously. The tests at STAR will focus on checking that the interfaces between DSM2 boards and other pieces of STAR hardware are functioning correctly. Testing of the remaining functions, including the final operational software, can happen during the shutdown

At the end of this test sequence all aspects of the generic DSM2 operation in STAR should be tested and working. The installation of boards into the specific crates where they will be used in Run22, and their programming with the actual algorithms that will be needed, will be covered in a separate plan.

VHDL Firmware Development

1. STP2C Interface

Implement and test (in loopback mode) the VHDL firmware to receive a Build_Event packet from STP2C and send back a data packet.

   a. I have examples of this from Chris's TRP project
   b. I need to coordinate with Chris to make sure I am using the same packet format as him for Build_Event commands and returning data packets
   c. Write VHDL to generate a Build_Event command internally, send it around a loop created by linking 2 of the DSM2's SFP+ sockets together and then send back an appropriate data packet containing at least some elements of the original Build_Event packet. Check that the received data packet matches the sent Build_Event command. Repeat.

2. Data Storage

Implement and test the VHDL firmware to read/write data to the SDRAM

3. Operational Firmware

Combine all the separate pieces of firmware to make the operational firmware including a module with well-defined interfaces and timing for the algorithms and places where future development of the SER functionality can be easily added.

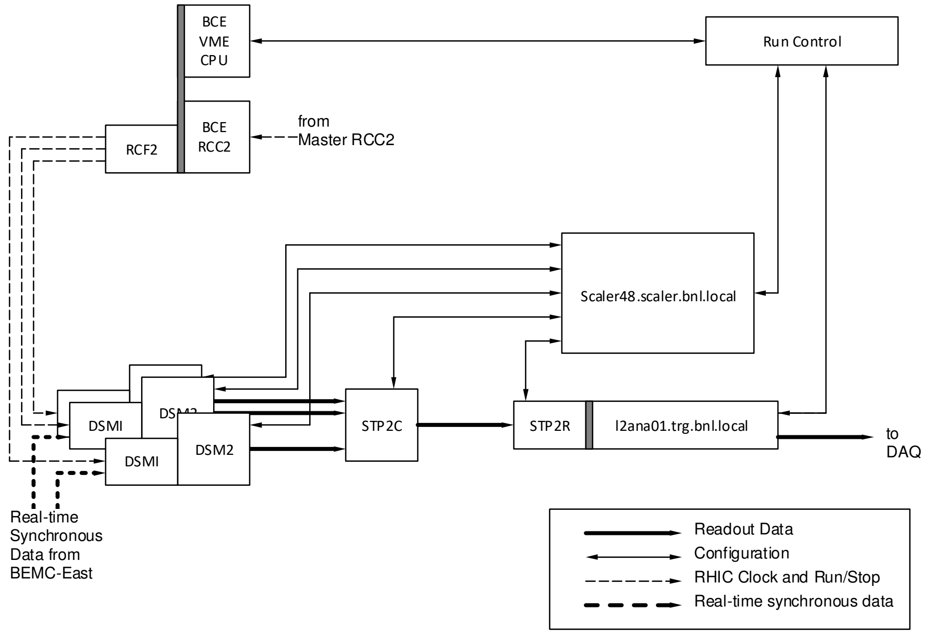
   a. I have early plans and diagrams for this that I need to resurrect.

Hardware Setup at STAR

DSM2 boards will be installed in the BCW crate for these tests. Any board in that crate receives data directly from BEMC, which is actually powered on and producing non-zero data. This will be essential for the TTL Input Timing Test.

For the purposes of these tests the DSM2 boards will be controlled from the scaler48 Linux computer on the scaler.bnl.local network. Scaler48 is also being used to control the STP2 boards. Physically each DSM2 board will be connected to the local switch using a standard Ethernet cable. The DSM2 boards will also be connected to the STP2C module in the BC1 crate via front panel fiber-optic cables. The connection to the BEMC detector will be through the existing back-of-crate interface cards.

The hardware setup for the Run21 tests of DSM2 boards is shown below.



Interface Testing at STAR

4. Non-Interference Test

Install multiple DSM2 boards in the middle of the CBLT of the BCW VME crate and check that those DSM2 boards do not interfere with the configuration or readout of the older DSM1 boards.

a. Three DSM2 boards, with the CBLT jumper ON, will be installed in slots 9, 10 and 11.
b. All DSM2 boards MUST have their sysreset-pullup jumper OFF.
c. I will create a DSM2 FPGA configuration that uses the RCC2 clock and Run/Stop signal by default. When the system is in Run mode the FPGA will latch in its 128 input bits from P3 and drive 32 bits out on P2, just like the DSM1 boards. The

DSM2 boards will therefore match the backplane activity of the DSM1 boards without needing any configuration.

d. The BCW VME CPU will configure all the remaining DSM1 boards as normal while skipping over the slots where a DSM1 has been replaced with a DSM2.
e. When the DSM1 boards are readout the CBLT should ignore or skip over the DSM2 boards. The BCW CPU will therefore receive a smaller-than-normal data block. Dummy data without any headers will be inserted in the trigger block wherever a DSM1 has been replaced with a DSM2.
f. A final stage would involve including this DSM crate in STAR data-taking for a few runs with beam. All normal STAR operations should work, with the only difference being that the final data block would contain sections of dummy data.

Status: All of a) through e) have been completed successfully. Part f) may not be necessary.

5. STP2 Connectivity Test

Connect multiple DSM2 boards to the STP2C board in the BC1 crate and check that the DSM2 boards correctly receive the Build_Event package and send back a verifiable data packet.

a. This test should involve the same DSM2 boards in the same DSM crate as before. All DSM2 boards should definitely now be connected to one STP2C board via front panel fibers.
b. As a first step I will create an FPGA configuration that instantiates multiple GTP link components with all of the necessary connections to enable monitoring of each link status, transmission of ping packets and word/frame counting of incoming packets.
c. The FPGA configuration will be loaded into the DSM2 boards. I will check that the link status bits are all correct (reset TXFAULT, MODDET, LOS, Lane-Up and Channel-Up) as seen on each DSM2 and on the STP2C.
d. The DSM2 port will have both TX and RX enabled and each DSM2 will have a unique NODE ID assigned to it. I will check that the STP2C monitor correctly receives the ping packets from all 3 DSM2 boards. I will check that each DSM2 board counts incoming words/frames with the source NODE ID of the STP2C.
e. For the next step I will create an FPGA configuration that waits until it receives a Build_Event command from STP2C, then creates a data packet containing some information from that command (e.g. token, DSM Address, Npre/Npost) and sends it back to the STP2C board as a data packet.
f. The STP2C board should receive packets from all DSM2 boards and pass them on to L2 via the STP2R.
g. L2 should just dump those packets in a file that is separate from the normal STAR data files.
h. This test is independent of the remaining DSM1 boards so it should not be necessary for them to be configured. They could be unplugged or the VME CPU could be booted in standalone mode (with the RCC2 left as a master) so that STAR Run/Stop commands would be ignored.
i. I can compare the streams of data from the DSM2 boards to each other. They should be identical. I can also compare them to the events stored in the normal

STAR data files. The token, DSM Address, etc… should be the same. I will need to write software to do this properly for a full run.

Status: Parts a) through d) have been completed successfully. Part e) is in progress. Parts f) to i) are still waiting.

6. RCC2 Synchronization Test

Check that the DSM2 boards correctly receive the RHIC clock and STAR Run/Stop signal and can maintain address synchronization with the existing TCU/DSM1/QT system.

a. This test should involve the same DSM2 boards connected to the same STP2C as before.
b. At least some of the DSM1 boards should be plugged in and the crate should therefore be included in Run Control so that the RCC2 and those DSM1 boards are properly configured.
c. I will modify the FPGA configuration to use the RHIC clock and control signals to make an address counter, just like all the other boards in the system. The logic will respond to the synchronous Latch command, which can be distributed by the RCC2 boards.
d. The logic will also save the current value of the counter in the data packets sent to STP2C.
e. As before the default values of the registers will be set so that the boards power-up in the correct configuration and further configuration should not be necessary.
f. Also as before the data packets will be passed from DSM2 to STP2C, to STP2R and on the L2, which will dump the data in a separate file.
g. The first stage of this test would involve using PedAsPhys runs when there is no beam. I would broadcast the Latch command via the RCC2, read the saved values from all the DSM1 and DSM2 boards in the crate and check that they remaining synchronized.
h. The second stage of this test would involve looking at the data files from L2. The data packet from each DSM2 board would now include the value of its address counter from the moment when the DSM2 received the Build_Event command. I can also compare those values to address counters read from other boards in the system, e.g. the TCU's DSM_Address value and the counter value read from each RCC2.

Status: Parts a), b) and c) are complete. Part d) is in progress and parts e) to h) are still waiting.

7. TTL Input Timing Test

Use a high-speed internal clock to measure the time difference between TTL input level changes and the rising edge of the RHIC clock for each input bit individually – or at least one bit from each input channel

a. This would be a test of a setup/monitoring scheme that should help with timing in new detectors and multi-layer trees.
b. The logic would start counting whenever there was a transition on an input bit and be saved/restarted by the rising edge of RHIC clock.
c. The user could latch the current values into static registers and read them out.

d. This test would need to have non-zero, changing data coming in from an outside source (BEMC?) and a fully configured RCC2 to provide the RHIC clock. It would not be necessary to have a STAR data taking run in progress.

Status: Not started.

Testing during the Shutdown

Details of this part still TBD.

8. Integration into Run Control

The user should be able to sit at the RC GUI, click DSM2 (crates?) in/out of the Trigger system, specify DSM2 register values and have the DSM2 boards properly configured at RUN_START time. The DSM2 “system” should send the appropriate response to RC for each command.

9. Integration into L2

The DSM2 data blocks must be properly placed into the TrgDataBlock with all the appropriate headers and pointers.

10. Communication with existing DSM Tree

DSM2 boards should be able to correctly pass data to downstream DSM/TCU boards and receive data from upstream, QT/DSM boards.

11. Full Operational Speed Tests

See how fast the system can run.

12. Real-time Monitoring

Make sure we can monitor the STP2C interface (counters, status bits, etc…) and the input timing automatically.