## **Testing Plan – 17-06-2025 – 24-06-2025**

### **Testing Plan Index**

**Duration:** 17-06-2025 to 24-06-2025

1. **Design Constraints – Important Notes**
2. **Checker Explanation – How the Testbench Works**  
      2.1. do\_analog\_capture\_checker(...)  
      2.2. read\_data\_from\_fifo\_checker(...)  
      2.3. sweep\_delay(...)  
      2.4. sweep\_delta(...)  
      2.5. write\_data\_to\_fifo(...) and read\_data\_from\_fifo(...)  
      2.6. do\_analog\_capture\_checker\_long(...) and read\_data\_from\_fifo\_checker\_long(...)  
      2.7. regression() and different\_clk()
3. **Testcase Plan**  
      3.1. Functional test summary  
      3.2. Clock configuration test summary

### **1. Design Constraints – Important Notes**

* **Slot Generation Logic:**
  + start\_slot comes when count = 0.
  + mid\_slot comes after start\_to\_mid\_delay - 1 cycles.
  + Next start\_slot comes after start\_to\_mid\_delay + mid\_to\_start\_delay - 1 cycles.
  + This cycle keeps repeating.
* **Key Rule:**
  + Once a start\_slot is given, the mid\_slot **must** wait until start\_to\_mid\_delay is complete.
  + You **can** delay the next start\_slot longer than expected – it’s allowed, and system will still work.
* **Programmable Delay Range:**
  + Both delays (start to mid and mid to start) are 5-bit.
  + Valid range: 24 to 31 (total 8 values possible).
* **Clock Domains:**
  + Define clock relations clearly – timing\_clk, ctrl\_clk, adc\_clk, and ana\_clk.
  + Make sure signals from different clocks are synchronized.

### **2. Checker Explanation – How the Testbench Works**

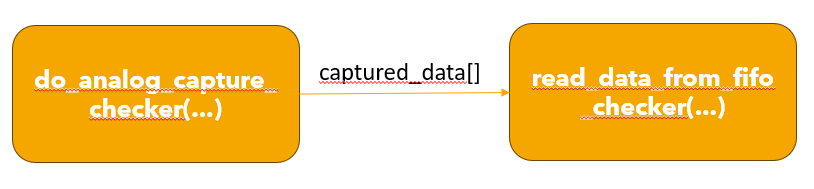
The testbench is structured using modular tasks to verify specific aspects of the design’s behavior, including data correctness, delay handling, and multi-clock domain interaction. Below is a detailed explanation of how each checker operates and what it validates.

### **do\_analog\_capture\_checker(...)**

**Function**: Primary checker for validating ADC data correctness.

* Waits for start\_slot\_detected, triggers ADC, asserts adc\_eoc, and captures ADC data.
* Repeats the process for mid\_slot\_detected.
* Combines both ADC values as a pair ({adc1, adc2}) and stores them in captured\_data[] at the given index.

**Purpose**: Confirms that analog data capture at both time slots occurs correctly and is stored as expected.

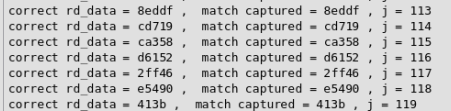
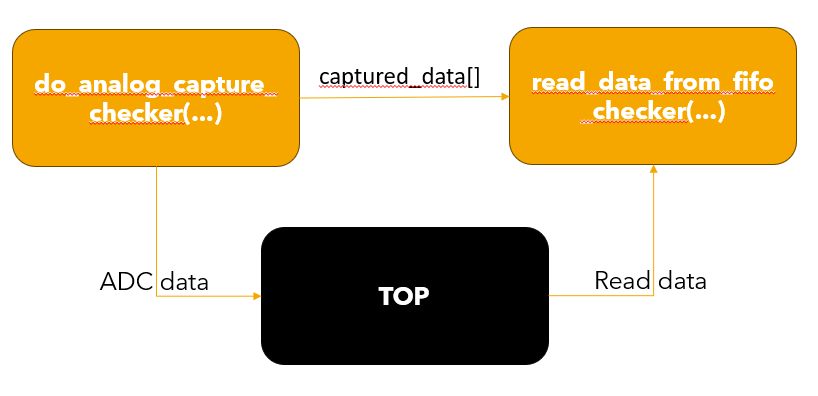


### **read\_data\_from\_fifo\_checker(...)**

**Function**: Verifies data integrity by reading from the FIFO.

* Enables FIFO read.
* Compares each rd\_data value with the corresponding entry in captured\_data[].
* Logs an error and increments the error counter if there is a mismatch.

**Purpose**: Ensures data written during ADC capture matches what is read from the FIFO.

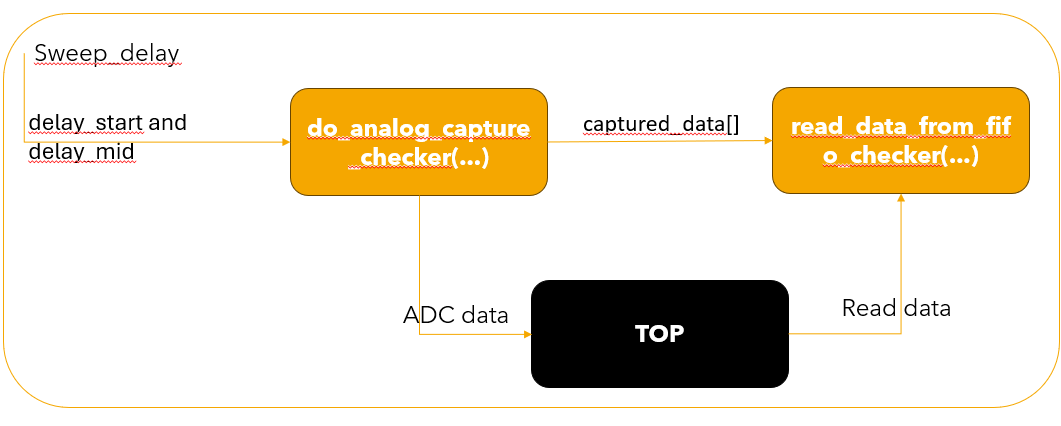


### **sweep\_delay(...)**

**Function**: Tests system behavior for a full range of programmable delay values.

* Iteratively varies delay\_start and delay\_mid across their allowable range (typically 19 to 31).
* Calls do\_analog\_capture\_checker for each combination.
* Validates collected data every 120 samples using read\_data\_from\_fifo\_checker.

**Purpose**: Ensures design stability and correctness across all valid programmable delay settings.

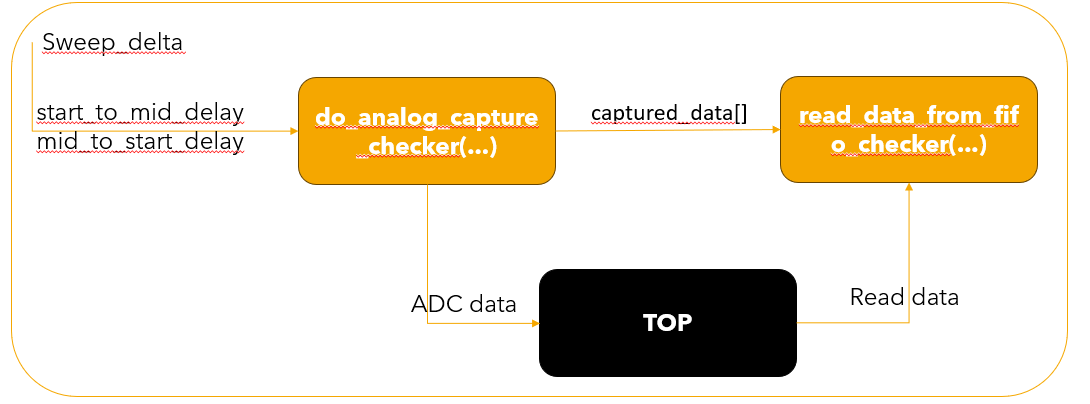


### **sweep\_delta(...)**

**Function**: Evaluates system behavior with varying slot-to-slot timing.

* Varies start\_to\_mid\_delay and mid\_to\_start\_delay. 32 to 255
* Checks whether the system maintains capture correctness across different slot intervals.

**Purpose**: Verifies that analog capture operates reliably with irregular or extended inter-slot delays.

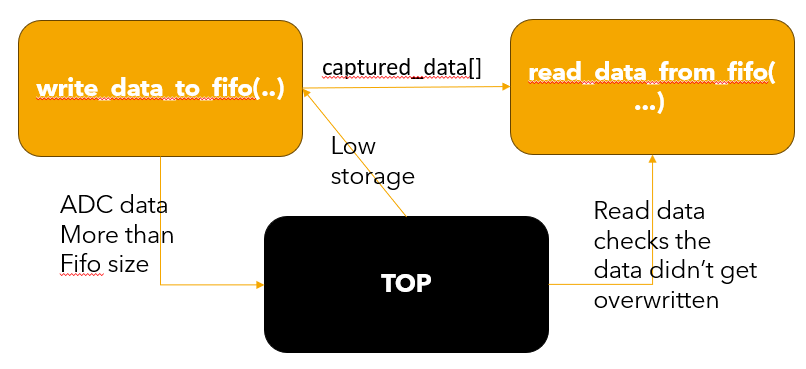


### **write\_data\_to\_fifo(...) and read\_data\_from\_fifo(...)**

**Function**: Conducts stress testing on FIFO operations.

* Writes a large number of ADC samples (e.g., 300) to the FIFO.
* Attempts to read and validate stored data.

**Purpose**: Tests FIFO depth handling, burst write and read operations, and overflow behavior.

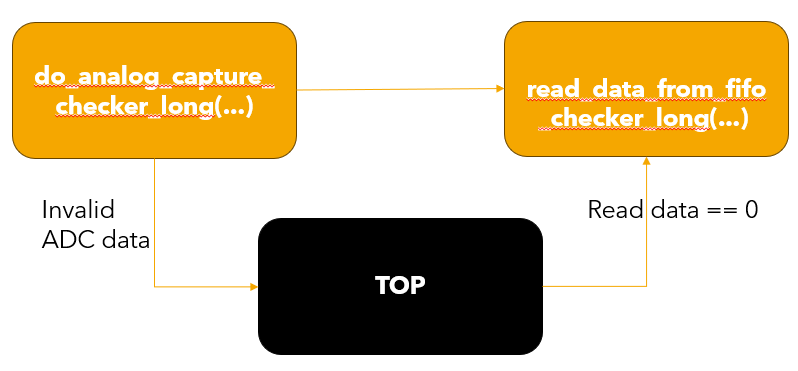


### **do\_analog\_capture\_checker\_long(...) and read\_data\_from\_fifo\_checker\_long(...)**

**Function**: Validates timeout conditions and analog delay handling.

* Uses the over\_max\_delay flag to simulate scenarios where analog delay exceeds the allowable limit.
* Ensures that such invalid samples are not written to FIFO.
* Checker expects to read zeros or invalid markers from the FIFO.

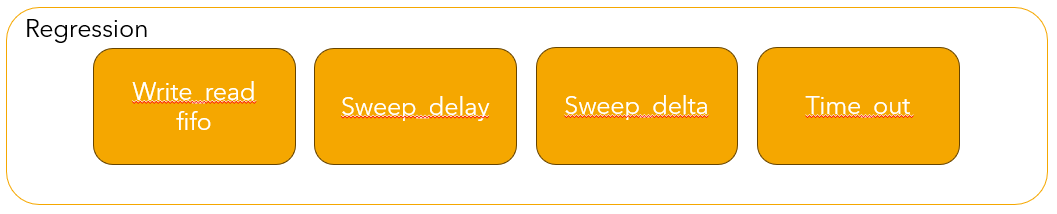
**Purpose**: Confirms the system’s ability to detect and reject analog data captured outside the valid timing window.



### **regression() and different\_clk()**

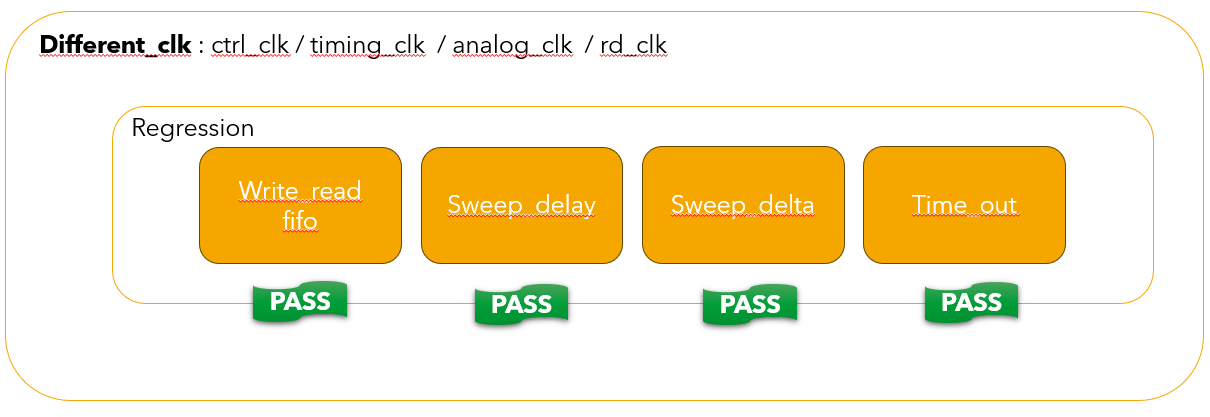
**Function**: Automated test suite to cover functional and timing corner cases.

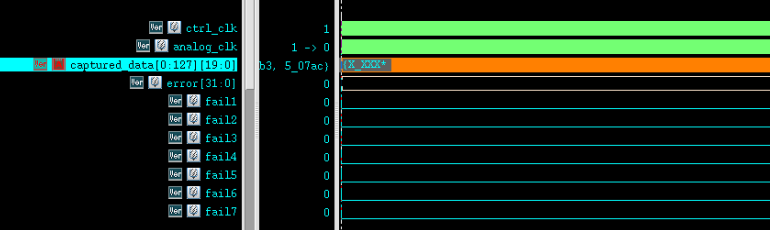
* regression() performs:
  + Basic data capture and verification
  + Delay sweeps
  + Timeout simulation
  + FIFO operation tests
  + Data integrity across read-write scenarios



* different\_clk() runs the entire regression suite across multiple clock configurations by varying:
  + ctrl\_clk
  + timing\_clk
  + analog\_clk
  + rd\_clk

**Purpose**: Validates the robustness of the design across a variety of clock domain interactions and ensures safe cross-domain behavior.





### **3. Testcase Plan**

### **24-06-2025 test results \* after Debugs**

|  |  |  |
| --- | --- | --- |
| **Testcase** | **What it Checks** | **Status** |
| Basic delay test | Normal function with fixed delays | Passed |
| Delay sweep | Try all delay values from 19 to 31 | Passed |
| Extra wait between slots | Delay next start\_slot more than required | Passed |
| Long analog delay | If analog\_delay > 10, data should be missed or flagged | Passed |
| FIFO burst read | Continuous ADC read – check if data is stored properly | Passed |
| FIFO overflow | Make sure data isn't overwritten when FIFO is full | Passed |
| Different clock speeds | Run tests with different clk periods | Passed |
| Reset check | Give reset in between and see recovery | Passed |
| Read and Write | Check if the system can read and write together | Passed |
| Overwrite before Read | Check if the system should not overwrite the data if it is not read | Passed |

### **Run tests with different clk periods \* after Initial Debugs**

|  |  |  |
| --- | --- | --- |
| **Clock Configuration** | **What it checks** | **Status** |
| ctrl\_clk = #5 (100 MHz) timing\_clk = #4 (125 MHz) analog\_clk = #3 (166 MHz) rd\_clk = #8 (62.5 MHz) | **Nominal case** — Reference configuration | Passed |
| ctrl\_clk = #10 (50 MHz) others unchanged | **Slow controller** — Stress test long FSM periods | Passed |
| ctrl\_clk = #2 (250 MHz) others unchanged | **Faster controller** — Stress test short FSM periods | Passed |
| timing\_clk = #2 (250 MHz) others nominal | **Fast timing domain** — High-frequency start\_slot/mid\_slot generation | Passed |
| analog\_clk = #7 (71 MHz) others nominal | **Slow analog module** — Simulates longer analog processing | Passed |
| analog\_clk = #2 (250 MHz) others nominal | **Faster analog module** — Simulates shorter analog processing | Passed |
| rd\_clk = #2 (250 MHz) others nominal | **Fast read domain** — Aggressive FIFO emptying | Passed |
| rd\_clk = #10 (100 MHz) others nominal | **Slow read domain** — Delayed FIFO emptying | Passed |
| All clocks set to #5 (100 MHz) | **Synchronous clocks** — Edge case | Passed |