Test Plan for Circular FIFO

1.. Verification Requirements

a) **Verification levels:** As we don’t have individual blocks, we are testing for targeted design unit. Single chip (FIFO) is verified as single unit (Top block).

b) **Functions:**

FIFO_FULL: FIFO FULL is asserted when the buffer has reached maximum writes and if buffer is not full FIFO FULL is zero.

FIFO_EMPTY: FIFO EMPTY is asserted when the buffer is empty or when the buffer doesn’t have any valid elements.

RD_EN: when read_en is asserted, data is read from fifo. When read_en is zero, It means we are not reading data from fifo. (read pointer is incremented)

WR_EN: when write_en is asserted, data is written into fifo. When It is zero, data is not been written into fifo.(write pointer is incremented)

CLEAR: When clear is asserted, the last element of data is cleared, which means write_pointer is decremented.

RESET: When RESET is asserted, all the elements in the fifo is cleared and fifo_empty is set to high.

Error: Error logic will be triggered when following cases occur.

- The fifo is empty and we try to read
- The fifo is empty and we try to clear
- The fifo is empty and we try to read and clear
- The fifo is full and we try to write
- Commands to write and clear at the same time
- Any command to read, write, or clear when RESET is asserted

c) Specific tests and methods:

i. Type of Verification: Black box testing.

ii. Verification Strategy:

We are using directed test cases to test the design. The verification environment of fifo has mainly RESET_driver and stimulus generation(main_sequence) and monitor. RESET_driver resets the fifo and asserts fifo_empty to high then main_sequence will generate the stimulus by reading and writing values into fifo.

The verification strategy mainly has scoreboards, checkers, stimuli, drivers, and monitors.

iii. Abstraction Level:

Fifo verification is done at top level. Outputs of fifo such as fifo_full, fifo_empty and read_data is verified with directed test cases. We have also covered the corner cases where we write interesting test cases like trying to write into fifo even if it is full …etc.

iv. Checking: Implemented Reference model checking style and assertions to check correctness of the design.

Reference model checking:

Implemented reference model called ref_model(task) and compared actual values and expected values in ref_model task.

Checker_fifo class has task called ref_model which has reference model logic.

We have used an inbuilt function ($) to declare fifo.
We used the following fifo functions :

1.push_back() → to write data.
2.pop_front() —> to read data.
3.pop_back() —> to clear data.

Inputs: rd_en,wr_en,RESET, wr_data,clear;
Outputs: queue_full, queue_empty, rd_data_ex

We used the black box approach for the reference model.

In the reference model task itself, we implemented a checker where the output (rd_data)of rtl design and (rd_data)reference model output are compared.

vif1.rd_data(rtl output) == rd_data_ex(reference model output)

If the read data of the fifo design and Reference model don't match then we display the expected data and actual data.

Features we test using reference model and assertions:

Reading data to fifo: This feature is verified using reference model checking where we compare rd_value and rd_value_ex.
ReadData assertion is also used to check if the rd_data is in the range to the FIFO read values.

Writing data to fifo: This design feature is verified using reference model checking where we compare rd_value which is taken from fifo and rd_value_ex. which is taken from FIFO queue.
WriteData assertion is also used to check if the write_data is in the range to the fifo values.

Clear data in fifo: This is tested using a reference model where we get value using pop_back().

Reset: It is tested using both the reference model and ResetFIFO assertion.

FIFOFULL: This is tested using both reference model and assertion(FifoFull).

FIFOEMPTY: This is tested using both the reference model and assertion(FifoEmpty).

Error logic: Assertions are written to test error logic.

Assertions:

We wrote concurrent assertions to check the correctness of the design.

Following assertions are written in RTL:

1.BypassLogic: assertion to check if rd_data is equal to write data when both read enable and write enable are asserted.

2.ReadData; This assertion checks if the value that is read is within range of fifo.

3. WriteData: This assertion checks if value of write data is within range of fifo.

4. ResetFIFO: assertion to check if Fifo is empty when Fifo is reset

5. FifoFull: This assertion checks if FIFO full is high when FIFO is full.

6.FifoEmpty.: assertion to check if fifo empty is high when fifo is empty.

7. Error logic1: "Fifo Empty and rd_en is asserted and error is high.
8. Error logic2: Fifo Empty and clear is asserted then error is high.
9.Error logic3: Fifo full and wr_en is asserted then error is high"
10. Error logic4:Fifo Empty,rd_en and clear is asserted then error is seen
11. Error logic5: "Wr_en and clear is asserted then error should be high.
12 Error logic6: error is high when Reset and either of wr_en,rd_en and clear is asserted.

d) **Coverage:** we have written coverage for inputs and outputs.

INPUTS: RESET, rd_en, wr_en, clear, wr_data.
OUTPUT: fifo_full, fifo_empty, rd_data,error

For different **Scenarios** that we considered below, we have created cover points in a cover group called “cg”.

The functional coverage that we achieved is100%.

e) Scenarios:

1. what happens if we insert data even after FIFO is full? [Error logic]

   Condition: fifo_full = 1,wr_en = 1;

2. what happens if we try to remove data even after FIFO is empty? [Error logic]

   Condition : fifo_empty = 1; rd_en = 1;

3. what happens if we clear FIFO even after FIFO is empty?

   Condition : clear =1; fifo_empty = 1;

4. what happens if you read and write asserts at a time?

   Condition: rd_en = 1; wr_en = 1;

5. what happens if clear and write arrest at the same time?

   Condition: clear = 1; wr_en = 1;

6. what happens when reset is asserted when fifo is empty?

   Condition: RESET =1; fifo_empty = 1;

7. what happens when reset is asserted when FIFO is full?

   Condition: RESET = 1; fifo_full = 1;

8. Check if the read data is valid. (data is said to be valid if it is within [0:SIZE-1] range)

Condition: rd_en = 1;

9. Check if write data is valid.

Condition : wr_en = 1;

10. What happens if reset and clear are asserted simultaneously?

Condition : RESET = 1; clear = 1;

11. What happens when read enable and write enable are asserted when fifo is empty?

Condition : rd_en = 1; wr_en = 1; fifo_empty = 1;

12. What happens when read enable and write enable are asserted when FIFO is full?

Condition : rd_en = 1; wr_en = 1; fifo_full = 1;

13. When fifo is reset and we try to write [Error logic]
RESET = 1, wr_en = 1;

14.When fifo is empty and we try to read and clear [Error logic]
Fifo_empty = 1, rd_en = 1, clear = 1;

15. Commands to write and clear at the same time [Error logic]
Wr_en = 1 and clear = 1;

16.Checking for corner test cases for write data = hFF,h00,hAA,h55.

II. Project Management

A. tools : Tools we have used to perform design and testing are Questasim by mentor graphics . Licence for Questasim is provided by MCECS Portland state university.

B. risks/dependencies:

Covering all test cases is always challenging, So we a team of two engineers tried to cover all the test cases. At this point of time we don't see any risks that are associated.

C. resources

Two verification engineers and Questasim license with Two good configured laptops.

D. schedule

To construct the whole design and verification we got 10 weeks of time. We would be working on RTL design in first couple of weeks and then we will work on updating the design and constructing a test bench for different scenarios. Write coverage module and tried to achieve 100 percent for functional coverage.