Ain Shams University
Faculty of Engineering
Computer and Systems Department
Spring 2024

Assignment
Due Date: May 7th, 2024

FIFO (First-In-First-Out) Buffer

Objective:
To design a synchronous FIFO (First-In-First-Out) buffer with the following features:

- Variable depth
- Write and read pointers
- Full and empty flags
- Synchronous write and read operations

Specifications:
You have two parameters for defining the width of data to be written or read, and for specifying the depth of the FIFO Buffer.

parameter DATA_WIDTH = 8 (8 is the default data width)
parameter DEPTH = 16 (16 is the default FIFO Buffer depth)

Here are the inputs and the outputs:

```
input logic clk,                          // Clock signal
input logic rst_n,                        // Active low reset signal
input logic wr_en,                        // Write enable signal
input logic rd_en,                        // Read enable signal
input logic [DATA_WIDTH-1:0] wr_data,     // Data input for write operation
output logic [DATA_WIDTH-1:0] rd_data,    // Data output for read operation
output logic full,                        // Flag to indicate FIFO is full
output logic empty                        // Flag to indicate FIFO is empty
```

Functional Requirements:

1. FIFO should support a variable depth specified by the DEPTH parameter.
2. Write and read operations should be synchronous to the clock.
3. FIFO should have separate full and empty flags.
4. The FIFO should wrap around when it reaches the end of the buffer.
5. On reset, all internal pointers should be reset and the FIFO should be empty.

Design Guidelines:

- Use an array to implement the FIFO buffer.
- Implement separate pointers for write and read operations.
- Implement logic to set and clear full and empty flags based on the current state of the FIFO.
- Implement synchronous write and read operations using the wr_en and rd_en signals.

Example Testbench:

```
// Clock generation
always begin
  #5 clk = ~clk;
end

// Reset generation
initial begin
  rst_n = 0;
  #10 rst_n = 1;
  #20;

  // Test cases
  wr_en = 1;
  wr_data = 8'hAA;
  #10;
  wr_data = 8'hBB;
  #10;
  wr_data = 8'hCC;
  #10;
  wr_en = 0;
  rd_en = 1;
  #10;
  rd_en = 0;
  #10;
  wr_en = 1;
  wr_data = 8'hDD;
  #10;
  wr_en = 0;
  rd_en = 1;
  #10;
  rd_en = 0;
  #10;
  $finish;
end
```