First need to import ibex folder to prepare

- python3 --version (checks version)
  - sudo apt update
- sudo apt install python3-pip
- sudo wget https://github.com/lowRISC/ibex
- cd ibex-master/
- pip3 install -U -r python-requirements.txt
- sudo wget https://github.com/stnolting/riscv-gcc-prebuilt/releases/download/rv32i-4.0.0/riscv32-unknown-elf.gcc-12.1.0.tar.gz
- sudo tar -xvf riscv32-unknown-elf.gcc-12.1.0.tar.gz -C /usr/local/sbin
- cd riscv32-unknown-elf/
- apt-get install libelf-dev
- sudo apt-get install verilator
- sudo apt-get install libfl2
- sudo apt-get install libfl-dev
- sudo apt-get install zlibc zlib1g zlib1g-dev
- sudo apt-get install verilator
- sudo depmod
- vim .profile
- if [-d "$HOME/.local/bin/riscv32/bin"] ; then
  PATH="$HOME/.local/bin/riscv32/bin:$PATH"
- source .bashrc
- sudo apt-get install srecord
- sudo make -C examples/sw/simple_system/hello_test
- ./build/lowrisc_ibex_ibex_simple_system_0/sim-verilator/Vibex_simple_system [-t] --meminit=ram,./examples/sw/simple_system/hello_test/hello_test.elf -t
- riscv32-unknown-elf-objdump --disassemble ~/ibex/examples/sw/simple_system/hello_test/hello_test.elf > hello_test.asm

VERLOG

_____

```
`timescale 1ns / 1ps

module fifo_32bit_depth5 (
    input clk,
    input reset,
    input wr_en,
    input [31:0] wr_data,
    input rd_en,
    output reg [31:0] rd_data,
    output reg [2:0] count
);

parameter DEPTH = 5;
localparam ADDR_WIDTH = $clog2(DEPTH);

reg [ADDR_WIDTH-1:0] wr_ptr, rd_ptr;
reg [DEPTH-1:0] [31:0] mem;

always @(posedge clk) begin
    if (reset) begin
        wr_ptr <= 0;
        rd_ptr <= 0;
        count <= 0;
    end else begin
        if (wr_en && count < DEPTH) begin
            mem[wr_ptr] <= wr_data;
            wr_ptr <= wr_ptr + 1;
            count <= count + 1;
        end
        if (rd_en && count > 0) begin
```

```
            rd_data <= mem[rd_ptr];
            rd_ptr <= rd_ptr + 1;
            count <= count - 1;
        end
    end
end

endmodule
```

VERILOG WITH TEST BENCH

Verilog

```
`timescale 1ns / 1ps

module fifo_32bit_depth5_tb;

    // Instantiate the DUT
    fifo_32bit_depth5 dut (
        .clk(clk),
        .reset(reset),
        .wr_en(wr_en),
        .wr_data(wr_data),
        .rd_en(rd_en),
        .rd_data(rd_data),
        .count(count)
    );

    // Clock and reset generation
    reg clk;
    reg reset;
    initial begin
        clk = 0;
        forever #5 clk = ~clk;
    end
    initial begin
        reset = 1;
        #10 reset = 0;
    end

    // Testbench signals
```

```
reg wr_en;
reg [31:0] wr_data;
reg rd_en;
wire [31:0] rd_data;
wire [2:0] count;

// Testbench logic
integer i;

initial begin
    // Reset the FIFO
    wr_en = 0;
    rd_en = 0;
    #20;

    // Write some data
    for (i = 0; i < 5; i++) begin
        wr_data = i;
        wr_en = 1;
        #10;
        wr_en = 0;
        #10;
    end

    // Read some data
    for (i = 0; i < 3; i++) begin
        rd_en = 1;
        #10;
        rd_en = 0;
        #10;
    end

    // Write some more data
    for (i = 3; i < 7; i++) begin
        wr_data = i;
        wr_en = 1;
        #10;
        wr_en = 0;
        #10;
    end

    // Read all data
    for (i = 0; i < 5; i++) begin
        rd_en = 1;
```

```
            #10;
            rd_en = 0;
            #10;
        end

        // Check if FIFO is empty
        #10;
        if (count !== 3'b000) begin
            $display("Error: FIFO not empty at end of test");
        end else begin
            $display("FIFO test passed");
        end

        $finish;
    end

endmodule
```