2024 Digital IC Design Homework II

NAME	洪緯宸
Student ID	M18121510

Functional Simulation Result

FIFO Pass	LIFO Pass	CIPU Pass

Stage 1

```
#
# There are total    0 errors in FIFO  !!
#
```

Stage 2

```
#
# There are total    0 errors in LIFO  !!
#
```

Stage 3

```
#
# There are total    0 errors in FIFO2 !!
#

There are total    0 errors in LIFO  !!

There are total    0 errors in FIFO  !!


There are total    0 errors in FIFO2 !!


        ****************************
        **                        **       |__||
        **  Congratulations!!     **      / O.O  |
        **                        **    /_____   |
        **  Simulation PASS!!     **   /^ ^ ^ \  |
        **                        **  |^ ^ ^ ^ |w|
        ****************************   \m___m__|_|

        Correct / Total :   100 / 100
```

Description of your design

Module Interface

Inputs:

clk: Clock signal to synchronize the operations of the module.

rst: Reset signal to initialize the module state and internal registers.

people_thing_in: 8-bit input, potentially representing character data about people.

ready_fifo, ready_lifo: Signals to indicate readiness to operate on FIFO and LIFO structures respectively.

thing_in: Another 8-bit input representing character data about things.

thing_num: A 4-bit input possibly used for indexing or specifying a count related to thing_in.

Outputs:

valid_fifo, valid_lifo, valid_fifo2: Validation signals that indicate valid data is being output from FIFO and LIFO structures.

people_thing_out, thing_out: 8-bit outputs for processed data from FIFO and LIFO.

done_thing, done_fifo, done_lifo, done_fifo2: Signals to indicate completion of operations on FIFO and LIFO.

Internal Logic and State Machine

The module employs a state machine with the following states defined:

IDLE: The default state waiting for input.

READ: State to read and buffer inputs into FIFO or LIFO.

FIFO_OUT: State to output data from the FIFO.

LIFO_OUT: State to output data from the LIFO.

FINISH: Terminal state indicating all processing is complete.

State Transitions and Operations

Initialization and Resets:

On reset (rst), all internal registers like fifo_head, fifo_tail, lifo_top, lifo_tail, control flags (valid_fifo, done_fifo, etc.), and internal buffers are initialized to their default states.

Reading Inputs:

In the READ state, the circuit checks the character range of people_thing_in to decide if it should store the data in the FIFO (if it's an uppercase letter). Simultaneously, it checks thing_in for special characters (;, $) to control operations related to the LIFO.

It uses thing_in to load into the LIFO unless it's a separator (;) or termination symbol ($). If a separator is detected and thing_num is zero, it sets up a count

(thing_num_buf) which will control the retrieval of elements from LIFO in subsequent cycles.

FIFO and LIFO Outputs:

In the FIFO_OUT state, data is sequentially read from passenger_fifo based on fifo_head and output until fifo_head equals fifo_tail, indicating the FIFO is empty.

In the LIFO_OUT state, data is read from the baggage_lifo from the bottom up (lifo_tail), outputting until lifo_top equals lifo_tail.

Handling Completion and Flags:

The circuit uses several flags to manage control flow and signal when outputs are valid (valid_fifo, valid_lifo, etc.) and when all processing is complete for each structure (done_fifo, done_lifo, etc.).

Summary

The design is structured to manage two types of data inputs and process them independently in FIFO and LIFO manners, providing flexibility in handling and order-sensitive data processing. Special characters in inputs dictate the flow control, making it highly dependent on the format and sequencing of incoming data. This type of module would be particularly useful in systems requiring structured data management, such as packet processing systems, inventory management systems, or complex computational pipelines in hardware accelerators.