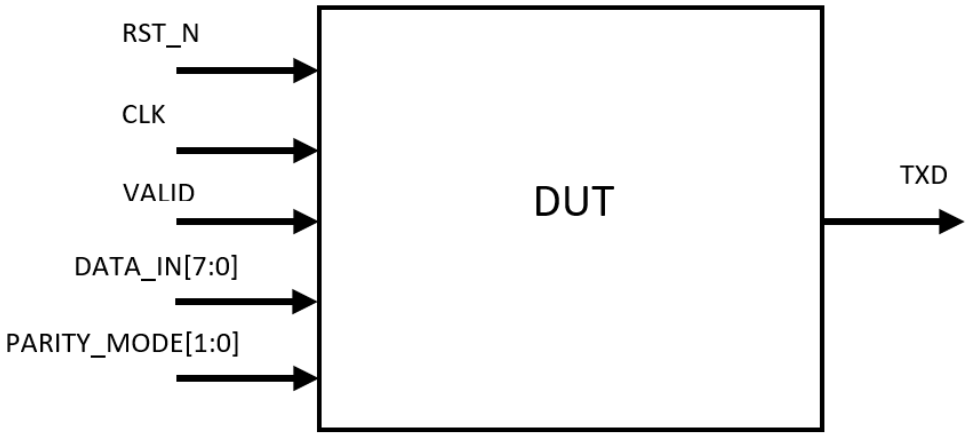
DV Advanced Entrance Test

I. Theory

1. What is ASIC design flow?
2. Can you explain a little bit about each step in the ASIC design flow?
3. What are the inputs for DV engineers?
4. What is the activity of DV after getting input from the RTL designer?
5. What is DV engineer responsibility?

(Refer to "Your task" on page 4 to see where to place the answers to those questions)

II. Practice



DUT (Design Under Test) is a converter module that has following features

- Convert an 8-bit data signal input into an 1-bit data signal output (parallel to serial converter).
- The 8-bit data signal (DATA_IN[7:0]) is only valid when VALID signal is High.
- The 1-bit data output signal can contain a parity bit based on input PARITY_MODE[1:0] configuration.

Port list	Direction	Description
RST_N	Input	Asynchronous reset (active low)
CLK	Input	IP clock (50MHz)
VALID	Input	Data input valid 0: DATA_IN is not valid 1: DATA_IN is valid
DATA_IN[7:0]	Input	8-bit parallel data input. This signal is valid only when VALID is 1
PARITY_MODE[1:0]	Input	Parity mode signal 2'b00,2'b11: No parity bit is inserted into output signal 2'b01: Odd parity bit is inserted into output signal 2'b10: Even parity bit is inserted into output signal
TXD	Output	Serial data out

Explanation on parity:

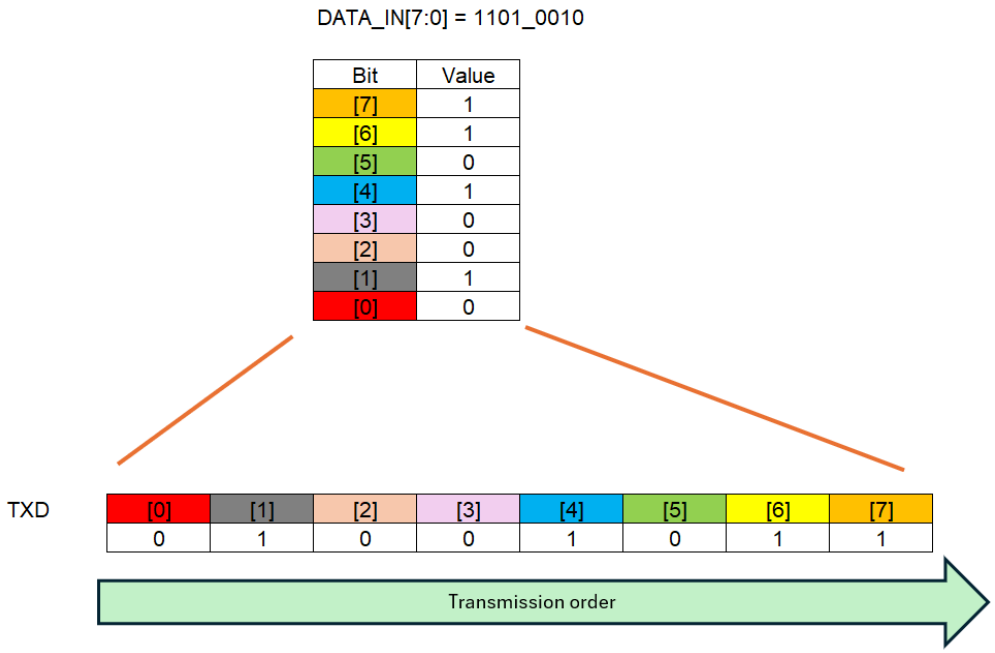
In data transmission, "parity" is a method used to check for errors in the data being transmitted. There are two main types of parity: even parity and odd parity

Even parity: The goal is to ensure that the total number of '1' bits in the data plus the parity bit is even. For example: If the data is `1010001` *(which has 3 '1' bits), the parity bit will be '1' (because 3 + 1 = 4, which is even). The data string will become* `10100011`*.*

Odd parity: The goal is to ensure that the total number of '1' bits in the data plus the parity bit is odd. For example: If the data is `1010001` *(which has 3 '1' bits), the parity bit will be '0' (because 3 is already odd, so no additional '1' bit is needed). The data string will become* `10100010`

Explanation on the serial transmission on TXD when the DATA_IN[7:0] is valid.

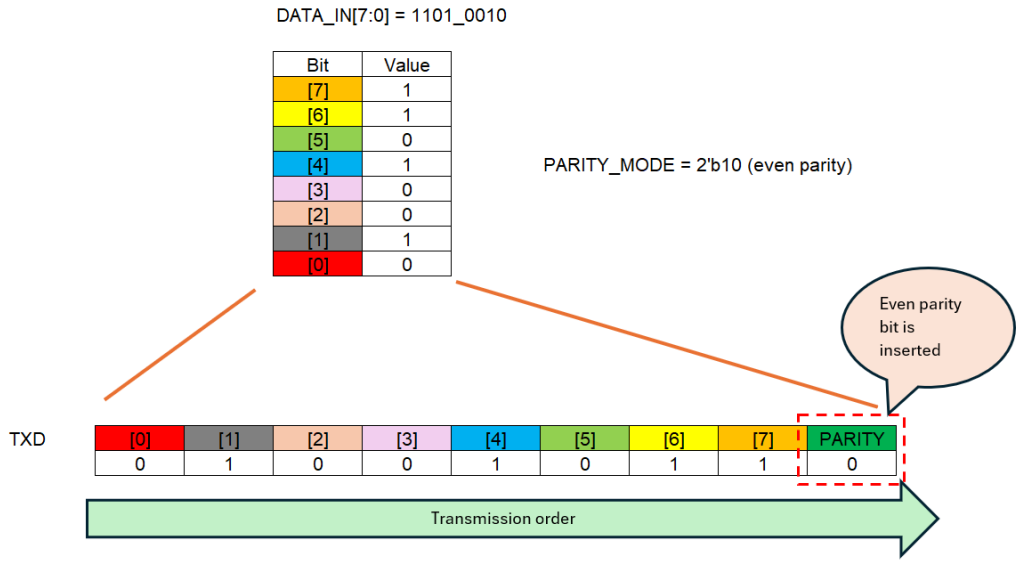
The LSB will be transmitted first.



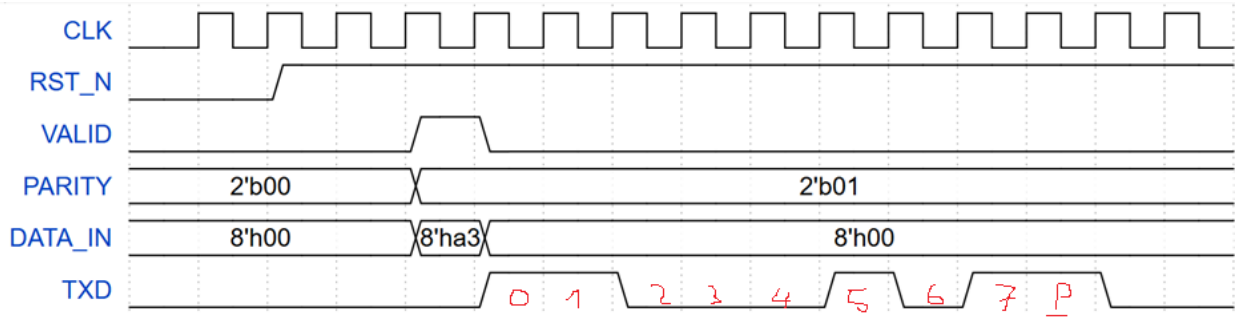
Explanation on PARITY_MODE[1:0]

PARITY_MODE[1:0] = 2'b00 or 2'b11: no parity bit is inserted to the output

PARITY_MODE[1:0] = 2'b01 or 2'b10: the parity bit is inserted at last bit on the TXD. The below example describe an example when PARITY_MODE = 2'b10.



Example: DATA_IN = 8'ha3, PARITY = 2'b01.



Your tasks:

Verify the correctness of this IP following below structure:

```
| -- practice
|   |-- docs                  // Vplan and theory answer
|   |   |-- theory_test.txt   // Answer question 1-5 here
|   |   |-- vplan.txt         // Verification plan.
|   |-- rtl                   // Design directory
|   |   |-- top.v             // RTL file
|   |-- sim                   // Simulation directory
|   |   | -- rtl.f            // RTL list
|   |   | -- tb.f             // Testbench list
|   |   | -- compile.f        // Compile list
|   |   | -- Makefile         // Run script list
|   |-- tb                    // Test bench directory
|   |   | -- test_bench.v     // Test bench file
```

a. Copy ictc/seminar-data/share_from_huy/dv_test/practice to your working directory.
b. Answer the question into file {your directory}/practice/docs/theory_test.txt
c. Create Verification Plan into file {your directory}/practice/docs/vplan.txt, list out all cases that need to be tested.
d. Go to tb/test_bench.v to create your testbench.
e. Create a Stimulus (input for the IP) and checker (to check the correctness of output) according to Verification Plan to verify this IP. (**NOTE**: There are some bugs in the RTL. Please document the bugs you find in the Verification Plan)
f. Refer the README in sim folder to execute the simulation and debugging.