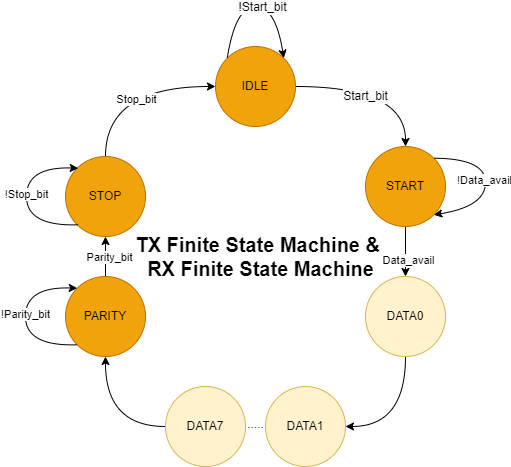
Capstone Project 2

Week 4 – Semester 233

Student: Vo Viet Hung – 2051076

**Implementation of AHB-APB and APB-UART bridges**

# The problem: TX FSM & RX FSM

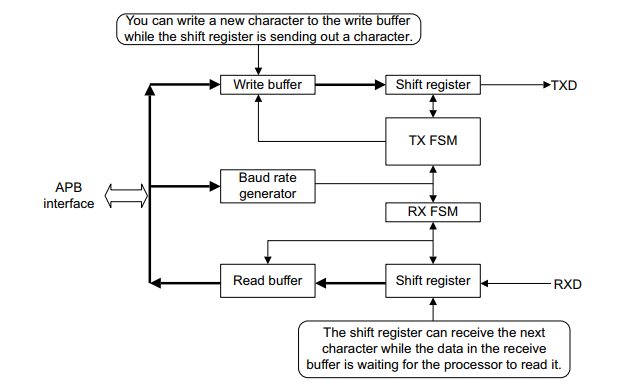
**1.1. Summary**

***Figure 1.1: TX FSM & RX FSM***

In week 31, I focused on designing and testing the finite state machines (**FSM**) for the transmitter (**TX FSM**) and receiver **(RX FSM**) of the **UART APB**.

Firstly, for the **TX FSM** & **RX FSM**, I identified the main states: **IDLE**, **START**, **DATA[0:7]**, **STOP**. I constructed a detailed state diagram and defined the state transition conditions based on control signals and the status of the **UART**. After finalizing the diagram, I wrote the **RTL** code and ensuring the synchronization and accuracy of each state.

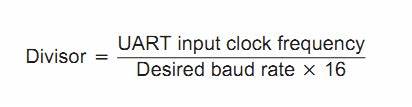
After completing the design, I set up a simulation environment to test both the TX FSM and RX FSM under various operating conditions. I created test such as inputting randomly data to check whether the data come out is correct or not, assuming no Parity\_bit, Start\_bit, Stop\_bit to ensure that the FSMs functioned correctly as designed. During this process, I debugged and optimized the code by generating more necessary signal to send back the signal come out from FSM for further combination with other process. Then, based on simulation and test results, I able to ensure that the FSMs operated efficiently and reliably.

**1.2. Solution**

***Figure 1.2.1: TX & RX references operation***

Firstly, I combine the previous jobs of week 29 and 30 together, so as you can see in the diagram, the operation will be from APB Interface, in which will generate the PWRITE signal, 32 bits data and others related to CPU (reset, clk). Then by assuming some random data come out from this I put them into UART operation.

Note: I have reviewed my design in previous weeks and adjusted for compatible.

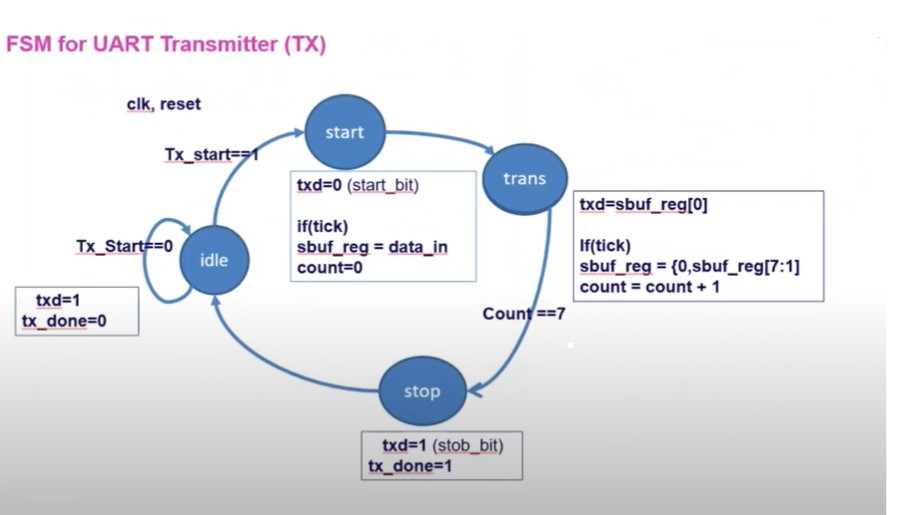
* Baud rate generator takes place as an internal clock generator for this block (named baud tick), so depend on the desired\_baud\_rate signal, it will create the UART clock pulse in compatible.

***Figure 1.2.2: Baudrate divisor references formula***

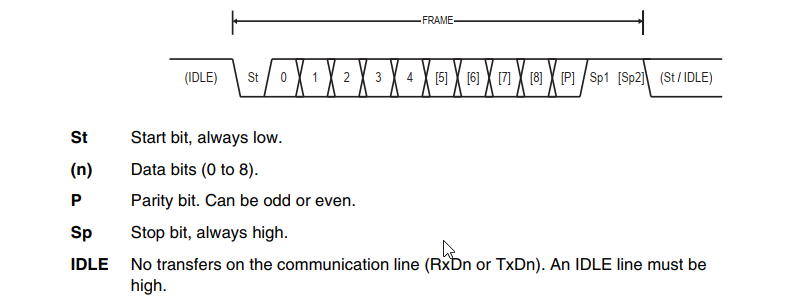
|  |  |  |
| --- | --- | --- |
| **No.** | **Desired Baudrate** | **Baudrate divisor** |
| 1. | 600 | 5208 |
| 2. | 4800 | 651 |
| 3. | 9600 | 325 |
| 4. | 19200 | 162 |
| 5. | 28800 | 108 |
| 6. | 38400 | 81 |
| 7. | 57600 | 54 |
| 8. | 115200 | 27 |
| 9. | 230400 | 13 |
| 10. | 460800 | 6 |
| 11. | 921600 | 3 |

***Table 1.2: Baudrate divisor references***

* TX – RX FSM are the state’s generator block for system to operate as its designs.
* Shift register is the block used for generating TXD and RXD signal and its operation depends mostly on FSM where it will receive the signal (send Start, Data, Parity, Stop bits) and feedback whether these signals are sent or not.
* Write – Read buffer are only used for storing data before requiring READ or WRITE data for make sure all data will not be missed during operation (Prevent Overlap data).

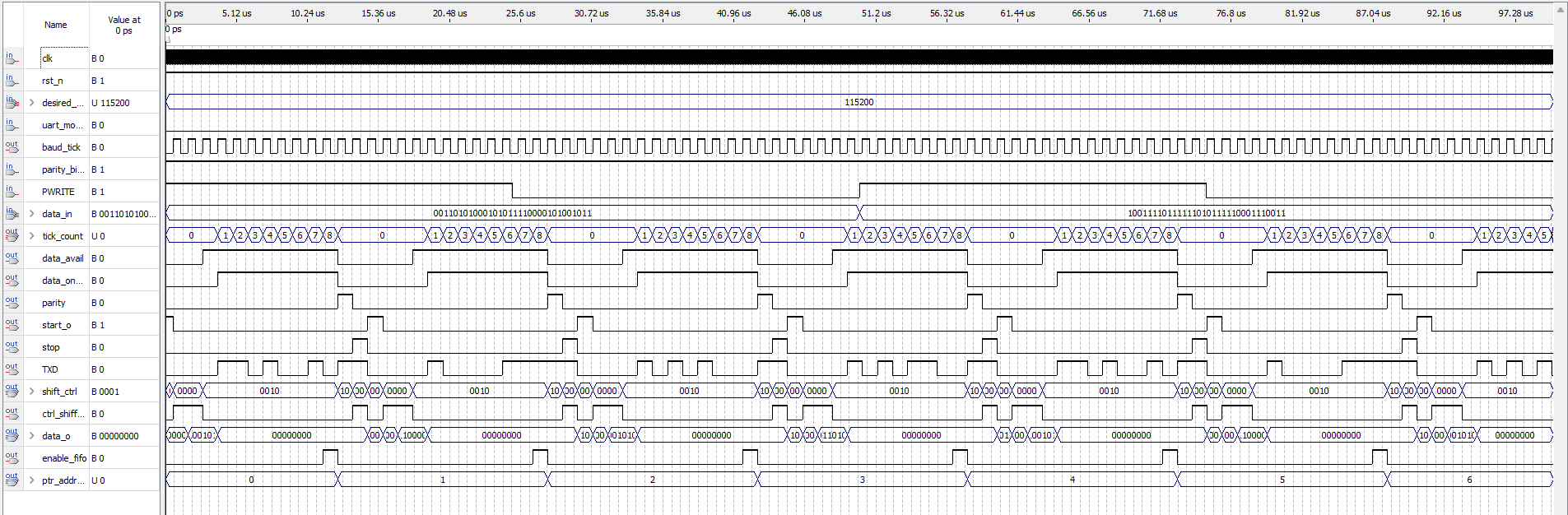
Because I already created the other block or their functionality is not require too much details so I will concentrate on FSM only.

***Figure 1.2.3: TX – RX FSM references***

Based on other design methodologies above ([Reference idea for TX and RX FSM](https://vn.images.search.yahoo.com/search/images;_ylt=Awrx_yPb3qdmvCkachdrUwx.;_ylu=Y29sbwNzZzMEcG9zAzEEdnRpZAMEc2VjA3BpdnM-?p=TX+FSM+UART&fr2=piv-web&type=E210VN91215G0&fr=mcafee#id=3&iurl=https%3A%2F%2Fuser-images.githubusercontent.com%2F106643865%2F199658638-16d30460-33a9-4ffe-82d4-aa939696fca9.jpg&action=click)) and UART’s operation in transmit and receive data frame below, I have concluded and regenerated my own FSM of TX and RX related to my capability.

***Figure 1.2.4: TX & RX data frame references***

Additionally, I did spent additionally times with implemented FSM with its generated case, and its signals to ensure data from APB Bridge through APB interface then inside UART generating accurate outputs. Finally, to me, the operate of this intermediate feature is fine.

Here is an example of TX simulation:

***Figure 1.2.5: Testing Operation of TX and RX UART***

***Explanation of ModelSim simulation:***

- For the first two upper signal, they are CPU clock and negative active reset generated from AHB to APB Bridge and pass into APB UART.  
- The next signal is desired\_baud\_rate indicates for the baud rate that the current APB UART is running on.

- Next, uart\_mode\_sel is the signal used for selecting which clock that APB UART is operating (without dividing down 16 or exactly to APB Bridge generates out).

- For the fifth signal, baud\_tick is the signal generated by UART CLOCK DIVISOR based on uart\_mode\_sel (0) and the desired\_baud\_rate ratio.

- parity\_bit\_mode is the mode selection for parity to generating out (if parity\_bit\_mode is 0 => the parity\_bit out will be 0 also, and so on for the remain case).

- PWRITE is the signal comes out from APB interface FSM which is used to write data into buffer for sending.

- data\_in[31:0] is the data comes out from APB interface also.

- tick\_count is the signal (customed) used for indicating number of bits have been sent.

- data\_avail is the flag bit used for indicating whether the current data (divided into 8 bits frame is available for sending or not).

- data\_on\_trans is the flag used for indicating that the data is transferred.

- parity is the parity bit reflect from a sending method.

- start\_o is the start bit reflect from a sending method.

- stop is the stop bit reflect from a sending method.

- TXD is the signal output of UART Transmission.

- shift\_ctrl[3:0] is the signal comes out from TX FSM for allowing shift register to send next bits.

- data\_o[7:0] is a customed signal for me to know which current data is transferred.

- enable\_fifo is a signal feedback from TX FSM, used for indicating TX Buffer sent out next data\_o[7:0].

- ptr\_addr\_rd is the signal used for showing that current data index is used.

# 2. Problems arise while I’m doing the project

# 2.1. Problem 1: WRITE – READ Buffer

Currently, I am able to send and read basic data that I generated randomly into APB interface so that will send out the exactly data to APB UART. However, to some extend that I have not figured out, the data of pointer address is struggled with a tiny problem where sometimes it keeps It previous data instead of increasing one. Therefore, it leads to a case that the current data is transfer is wrong or not able to send out the next 8 bits data out of Buffer.

**2.2. Proposed Solution:**

I have an idea that I will create another MUX (same as PC counter in RISC-V CPU) for selecting whether the next ptr\_addr\_rd or ptr\_addr\_wr will come into the buffer, by that way, I can completely control the pointer and easier to debug.

# 3. Plan for next week working

As I am continuing to fixed the problem inside UART APB, I will try to combine the APB Bridge and APB UART I have created together and simulate some necessary data for showing my project is working.

Moreover, I have an extra idea that I will use more states (such as ERROR, CLEAR, VALIDATION) inside TX – RX FSM for more clearly operation because during my test cases, I have seen that the states wrong at some tests. Until this moment, I can conclude that currently, my project’s progress is still on smooth. The main reason that I cannot come to University for asking tutorial is because I struggle with my internship so I hope you will not neglect my Journal. If I meet any troubles, I will ask you later.

**Thanks for reading my Journal.**

# References

1. *AMBA APB Protocol Specification,* <https://developer.arm.com/documentation/ihi0024/latest/>
2. *Arm Cortex-M System Design Kit Technical Reference Manual r1p1,* <https://developer.arm.com/documentation/ddi0479/d/apb-components/apb-uart>
3. Lakshmisagar H.S1, Sumathi M.S2 1,2Assistant Professors, BMS Institute of Technology, Bangalore, Karnataka, India, *Design and Verification of APB Compliant Quad Channel UART* <https://www.ijraset.com/fileserve.php?FID=9093>
4. A. K. Gupta, A. Raman, +1 author Ravi Ranjan, *Design and Implementation of High-Speed Universal Asynchronous Receiver and Transmitter (UART).* <https://www.semanticscholar.org/paper/Design-and-Implementation-of-High-Speed-Universal-Gupta-Raman/a8796177ab085bddf1836fda479085c66f3a3607>