Embedded Systems

Assignment-2

By:- Kwanit Gupta
B19EE046

Task:- To perform detailed modeling of interface protocol (AMBA APB)

1. According to the tasks assigned in respect to Roll-Numbers, my task is to **model different transfers in AMBA APB Protocol**.

2. For convenience and simplicity, my preference for coding language is **python**.

3. For the medium of inputs, it takes the help of ***Total Simulation Time*** (As defined by user and assumption hereby was taken that a clock width equals to some seconds-length), option as an integer value to perform either **Write Transfer without wait state** or **Read Transfer without wait state.**

```
Enter the Total Simulation Time :-24
Enter the Mode of Transfer amongst :-
 1.) Write Transfer without wait
 2.) Read Transfer without wait
```

4. Depending upon the choice of the user, the task can be initiated. But for each task, **an address** and **data (in numeric form for easier understanding)** become requirements for the demonstration and progress monitoring (they should be expressible within 32-bits configuration).

```
Enter the Address where data should be written (Expressible within 32 bits) :-1165
Enter the numeric data to be written (Expressible within 32 bits :-19521
```

5. The working of the model can be understood by taking assumptions as follows:-

a. All the Highs and Lows are considered as Boolean Values (True for High and False for Low)

b. The clock width and clock cycles are taken into seconds such that with each iteration, the progress of each signal could be understood alongside switching Highs and Lows with suitable time delays.

c. With the provided references, the modeling of each and every task was strictly done in accordance with T4 (4 cycles) and T6 (6 cycles) for Write Transfer without wait state and Read Transfer without wait state, respectively.

d. For the understanding of storing data and extracting data, binary lists were prepared for both the tasks, alongside the inverse function to convert binary data into output.

Regarding Test Bench, it was provided with all the signals set to LOWs (False) and Address alongside Data Buses to be Zero Arrays.

6. Output interface for Write Transfer without wait state and Read Transfer without wait state can be seen as following:-

```
At t=0 time-unit, PCLK=False | PWRITE=False | PSEL=False | PENABLE=False | PREADY=False |
At t=1 time-unit, PCLK=True | PWRITE=False | PSEL=False | PENABLE=False | PREADY=False |
At t=2 time-unit, PCLK=True | PWRITE=False | PSEL=False | PENABLE=False | PREADY=False |
At t=3 time-unit, PCLK=True | PWRITE=False | PSEL=False | PENABLE=False | PREADY=False |
At t=4 time-unit, PCLK=False | PWRITE=False | PSEL=False | PENABLE=False | PREADY=False |
At t=5 time-unit, PCLK=False | PWRITE=False | PSEL=False | PENABLE=False | PREADY=False |
At t=6 time-unit, PCLK=False | PWRITE=False | PSEL=False | PENABLE=False | PREADY=False |

 It's Time to write the data without wait state

Address before switch-on :- [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
Address after switch-on :- [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 1]

Data before switch-on :- 19521
Data after switch-on :- [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1]

At t=7 time-unit, PCLK=True | PWRITE=True | PSEL=True | PENABLE=False | PREADY=False |
At t=8 time-unit, PCLK=True | PWRITE=True | PSEL=True | PENABLE=False | PREADY=False |
At t=9 time-unit, PCLK=True | PWRITE=True | PSEL=True | PENABLE=False | PREADY=False |
At t=10 time-unit, PCLK=False | PWRITE=True | PSEL=True | PENABLE=False | PREADY=False |
At t=11 time-unit, PCLK=False | PWRITE=True | PSEL=True | PENABLE=False | PREADY=False |
At t=12 time-unit, PCLK=False | PWRITE=True | PSEL=True | PENABLE=False | PREADY=False |
At t=13 time-unit, PCLK=True | PWRITE=True | PSEL=True | PENABLE=True | PREADY=True |
At t=14 time-unit, PCLK=True | PWRITE=True | PSEL=True | PENABLE=False | PREADY=True |
At t=15 time-unit, PCLK=True | PWRITE=True | PSEL=True | PENABLE=True | PREADY=True |
At t=16 time-unit, PCLK=False | PWRITE=True | PSEL=True | PENABLE=True | PREADY=True |
At t=17 time-unit, PCLK=False | PWRITE=True | PSEL=True | PENABLE=True | PREADY=True |
At t=18 time-unit, PCLK=False | PWRITE=True | PSEL=True | PENABLE=True | PREADY=True |
At t=19 time-unit, PCLK=True | PWRITE=False | PSEL=False | PENABLE=False | PREADY=False |
At t=20 time-unit, PCLK=True | PWRITE=True | PSEL=True | PENABLE=True | PREADY=True |
At t=21 time-unit, PCLK=True | PWRITE=False | PSEL=False | PENABLE=False | PREADY=False |
At t=22 time-unit, PCLK=False | PWRITE=False | PSEL=False | PENABLE=False | PREADY=False |
At t=23 time-unit, PCLK=False | PWRITE=False | PSEL=False | PENABLE=False | PREADY=False |
At t=24 time-unit, PCLK=False | PWRITE=False | PSEL=False | PENABLE=False | PREADY=False |
```

Write Transfer without wait state

```
Enter the Address where data should be read (Expressible within 32 bits) :-1165
Enter the numeric data to be read (Expressible within 32 bits :-19521
At t=0 time-unit, PCLK=False | PWRITE=False | PSEL=False | PENABLE=False | PREADY=False |
At t=1 time-unit, PCLK=True | PWRITE=False | PSEL=False | PENABLE=False | PREADY=False |
At t=2 time-unit, PCLK=True | PWRITE=False | PSEL=False | PENABLE=False | PREADY=False |
At t=3 time-unit, PCLK=False | PWRITE=False | PSEL=False | PENABLE=False | PREADY=False |
At t=4 time-unit, PCLK=False | PWRITE=False | PSEL=False | PENABLE=False | PREADY=False |
At t=5 time-unit, PCLK=True | PWRITE=False | PSEL=True | PENABLE=False | PREADY=False |
At t=6 time-unit, PCLK=True | PWRITE=False | PSEL=True | PENABLE=False | PREADY=False |

 It's Time to read the address without wait state

Address before switch-on :- [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 1]
Address after switch-on :- 1165

At t=7 time-unit, PCLK=False | PWRITE=False | PSEL=True | PENABLE=False | PREADY=False |
At t=8 time-unit, PCLK=False | PWRITE=False | PSEL=True | PENABLE=False | PREADY=False |

 It's Time to read the data without wait state

Data before switch-on :- [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1]
Data after switch-on :- 19521

At t=9 time-unit, PCLK=True | PWRITE=False | PSEL=True | PENABLE=True | PREADY=True |
At t=10 time-unit, PCLK=True | PWRITE=False | PSEL=True | PENABLE=True | PREADY=True |
At t=11 time-unit, PCLK=False | PWRITE=False | PSEL=True | PENABLE=True | PREADY=True |
At t=12 time-unit, PCLK=False | PWRITE=False | PSEL=True | PENABLE=True | PREADY=True |
At t=13 time-unit, PCLK=True | PWRITE=True | PSEL=False | PENABLE=False | PREADY=False |
At t=14 time-unit, PCLK=True | PWRITE=False | PSEL=True | PENABLE=True | PREADY=True |
At t=15 time-unit, PCLK=False | PWRITE=False | PSEL=True | PENABLE=True | PREADY=True |
At t=16 time-unit, PCLK=False | PWRITE=False | PSEL=True | PENABLE=True | PREADY=True |
At t=17 time-unit, PCLK=True | PWRITE=False | PSEL=True | PENABLE=True | PREADY=True |
At t=18 time-unit, PCLK=True | PWRITE=False | PSEL=True | PENABLE=True | PREADY=True |
At t=19 time-unit, PCLK=False | PWRITE=False | PSEL=True | PENABLE=True | PREADY=True |
At t=20 time-unit, PCLK=False | PWRITE=False | PSEL=True | PENABLE=True | PREADY=True |
At t=21 time-unit, PCLK=True | PWRITE=False | PSEL=True | PENABLE=True | PREADY=True |
At t=22 time-unit, PCLK=True | PWRITE=False | PSEL=True | PENABLE=True | PREADY=True |
At t=23 time-unit, PCLK=False | PWRITE=False | PSEL=True | PENABLE=True | PREADY=True |
At t=24 time-unit, PCLK=False | PWRITE=False | PSEL=True | PENABLE=True | PREADY=True |
```

Read Transfer without wait state

According to the references and the necessary progress representations, these implementations worked in similar fashions.

7. According to the market/industry standards, implementation works in more complicated ways, since hereby many assumptions were taken into consideration for simplicity. Also, the way storage and signal-based communications were shown here, doesn't happen to high-end devices and systems at all, if seen with respect to market/industry standards.

So, on a concluding note, this implementation is fairly simple to understand and visualize, but not even an inch closer to being accurate in real-life devices.