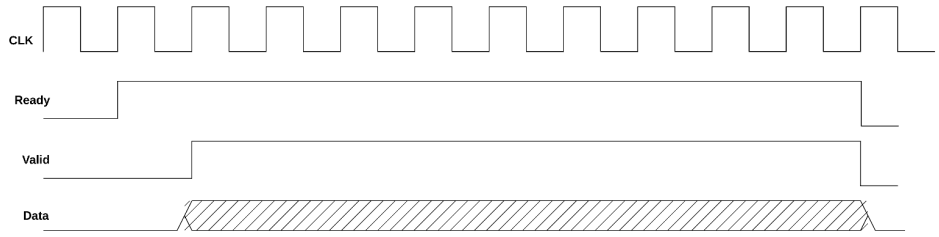
Design Details:
The `slave` module is a simple interface module that implements part of the **Valid-Ready Handshake Protocol** to control data transfer. It has two main inputs (`valid` and `data`) that drive data availability, and an output (`ready`) that signals the slave's readiness to accept data.

Module Description:

- **Inputs:**
  - `clk`: Clock signal for synchronizing logic.
  - `rst_n`: Active-low reset signal to initialize the `ready` signal.
  - `valid`: Input signal that indicates whether the `data` on the input is valid and ready for processing.
  - `data[3:0]`: 4-bit data bus containing the data to be transferred.
- **Output:**
  - `ready`: An output signal that indicates whether the `slave` module is ready to accept data.

Protocol Description:

Following a reset, the design asserts the `Ready` signal and keeps it high until 10 valid transactions (where `valid` is high and `data` is present) have been completed. After these 10 transactions, the `Ready` signal is de-asserted temporarily before reasserting and repeating the cycle. Timing diagram is below:



Testbench Task:

Add code into the driver and sequence class, utilizing the Slave sequence feature in UVM. Follow these steps to complete the task:

1. Monitor the `Ready` signal from the DUT interface and initiate a request to the Sequence when `Ready` is high.

2. Respond from the sequence by sending 10 `valid` transactions
3. After driving 10 transactions, reset the `valid` and `data` signals to zero on the following positive clock edge in the driver class.
4. Repeat the sequence 100 times.

Submit the code files along with wave screenshots.