Bigstream - Verilog Challenge-AXI4 Arbiter

Payman Behnam
Payman.behnam@utah.edu

In this brief, I summarize an explanation about the spec verilog implementation:

Like every other implementation, the code is based on my understanding of the specification. I used couple of references to understand the AXI4 and its hand-shaking mechanism [1,2,3].

For the arbiter implementation, there are different mechanisms including but not limited to Daisy chaining method, polling or rotating priority method, fixed priority or independent request, centralized or distributed. It can also be a combination of some of these approaches [4.5,6].

I considered the round-robin, non-preemptive implementation. This means that in the case of bus contenders, I use round-robin algorithm to assign bus to one of them. However, the bus will be released after the resource is finished with sending all the information (t_last =1). We can consider different scenario, but coding wise they follow the same principles.

I also wrote a test bench and tried different scenarios and coroner cases. Based on my best understating, it worked correctly.

Figure 1 shows an output of one possible scenario. Here I explain different situations.

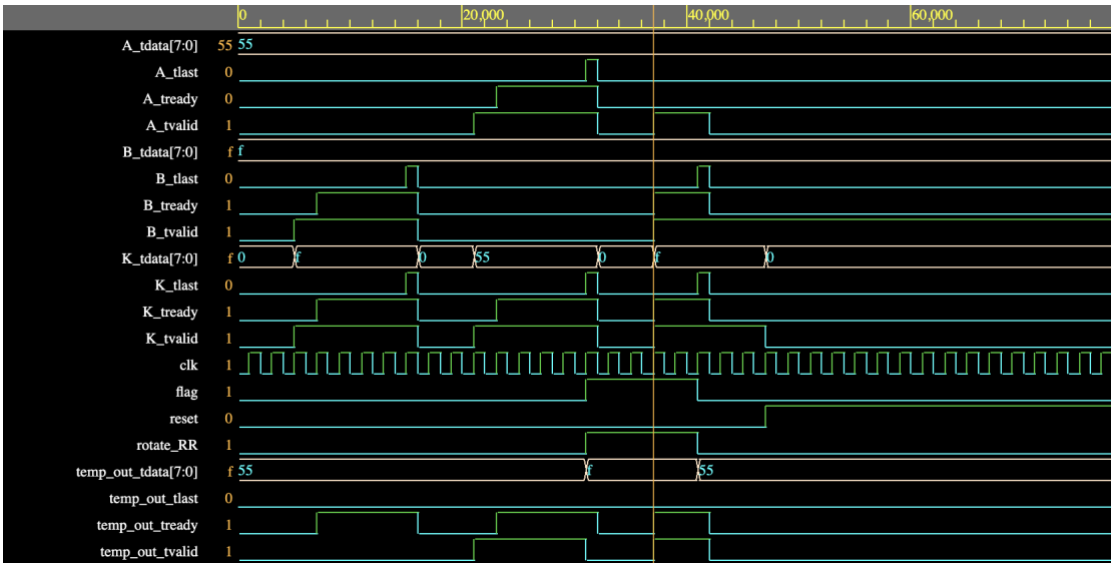

Figure 1- Waveforms of a simple AXI4 arbiter implementation.

- At the begging, none of A and B have valid values. So, all the outputs are zero.
- After some time, input B have valid value, but A doesn't have a valid one. Hence, the bus is assigned to B. The output K will reflect the value of input B. Note that output B_tready becomes '1' when K-tready becomes '1' as well.
- Later, A have valid value, but B doesn't have a valid value. Hence, the bus is assigned to A. The output K will reflect the value of input A. Note that output A_tready becomes '1' when K-tready becomes '1' .
- Afterwards, we have a case while both A and B have valid values. The round-robin mechanism chooses B. The output K will reflect the value of input B. Note that output B_tready becomes '1' when K-tready becomes '1' while the output A_tready is '0'.
- Next, the circuit is reset. Hence, all the outputs are '0' (some inputs such as B-tvalid still may have '1' or '0' value).

References:

[1] Lauri Võsandi, Arbitrary data streams, Available online: https://lauri.xn--vsandi-pxa.com/hdl/zynq/axi-stream.html
[2] Dillon Huff, AXI4principle, Available online: https://www.youtube.com/watch?v=lI5Gh-1zk-s
[3] AMBA specification, ARM Inc, Available online: https://www.arm.com/products/silicon-ip-system/embedded-system-design/amb specifications

[4] Sarangdhar, Nitin V., Konrad K. Lai, Gurbir Singh, Michael W. Rhodehamel, and Matthew A. Fisch. "Computer system with distributed bus arbitration scheme for symmetric and priority agents." U.S. Patent 5,581,782, issued December 3, 1996.
[5] Shah, Hardik, Andreas Raabe, and Alois Knoll. "Priority division: A high-speed shared-memory bus arbitration with bounded latency." In *2011 Design, Automation & Test in Europe*, pp. 1-4. IEEE, 2011.
[6] Pyoun, Chang Hee, Chi Ho Lin, Hi Seok Kim, and Jong Wha Chong. "The efficient bus arbitration scheme in SoC environment." In *The 3rd IEEE International Workshop on System-on-Chip for Real-Time Applications, 2003. Proceedings.*, pp. 311-315. IEEE, 2003.