

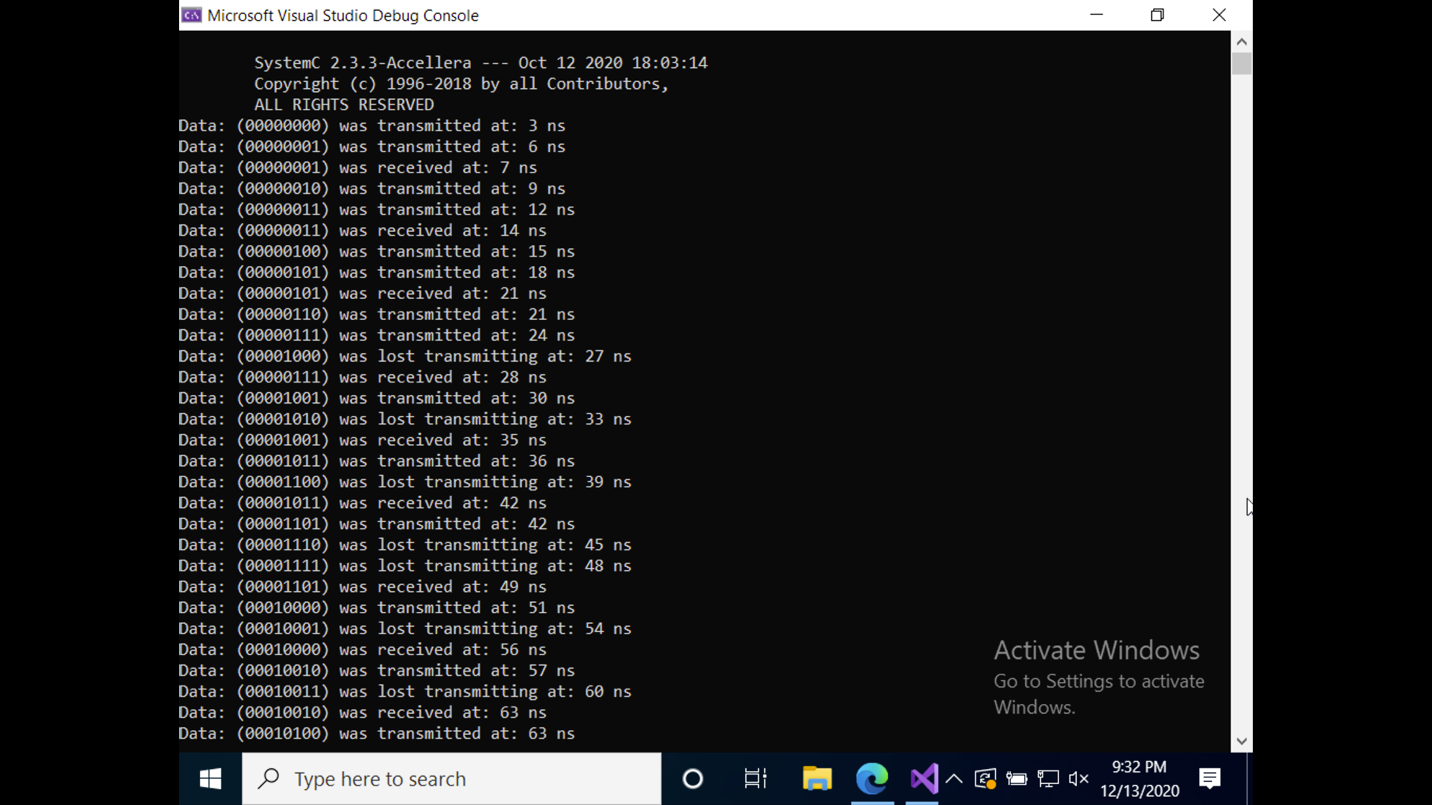
1. The systemC code and the testbench of the cosine approximation are included in the Final\_Q1 folder. The testing inputs are in radius, 0 < xin < 1.57. The best iteration number of this calculation is 2.

Graphical user interface, application

Description automatically generated

Figure 1. Test results of the cosine approximation calculation in SystemC. When xin was ‘0.5’ and when start was ‘0’, the calculation did not start. However, after start was ‘1’, the calculation successfully output 5 different inputs with the best iteration number of 2. When time was at 1802ns, the first iteration of cos(1.3) was incorrect, the second iteration of cos(1.3) was correct, which was 0.267499. When time was at 3026ns, the first iteration of cos(0.5) was incorrect, the second iteration of cos(0.5) was correct, which was 0.877583.

1. All the code are included inside of folder Final\_Q2. Question 2 implemented a non-blocking buffer channel using export on the target side.



Graphical user interface, text

Description automatically generated

Figure 2. This test results proves that this buffer has non-blocking channel. When data is failed to transmit or receive, it showed output error, such as at 27ns, 33ns, and 39ns.

1. 1. For read or write, the imitator send (forward) begin request, then goes into target, the target issues TLM accept (return path), the initiator now knows that the TLM has been accepted, and knows that the initiator wants to read or write.

After a delay, the target send (backward) end request to the initiator, because the target has completely seen a request and understood the request so the target is ready to go to the next step. Then the initiator issues the accept (return path).

After a delay, the target issues (backward) begin respond and the initiator issues accept from the return path to notify the target that the initiator is ready to read or write data.

After a delay, the data will be either read or written after the begin respond.

After a delay, and after the data has been successfully transmitted, the initiator will issue (forward) end respond to the target. The target will issue (backward) accepted to indicate that this transaction is completed. Shown in figure 3.

Text

Description automatically generated

Figure 3. Result of example 6 (nbFwdBwdMemoryRW: 4 calls) in the lecture series on TLM implements a non-blocking communication using 4 calls.

* 1. Set generic payload attributes to transfer 3 streams of data (instead of 5) in each transaction transportation between initiator and target, shown in Figure 4.

Text

Description automatically generated

Figure 4. Result of nbFwdBwdMemoryRW (4 calls) which changed the generic payload to transfer 3 streams of data instead of 5 streams of data.

* 1. The code is implemented.
  2. The timing has changed.
  3. Analyze the new result.

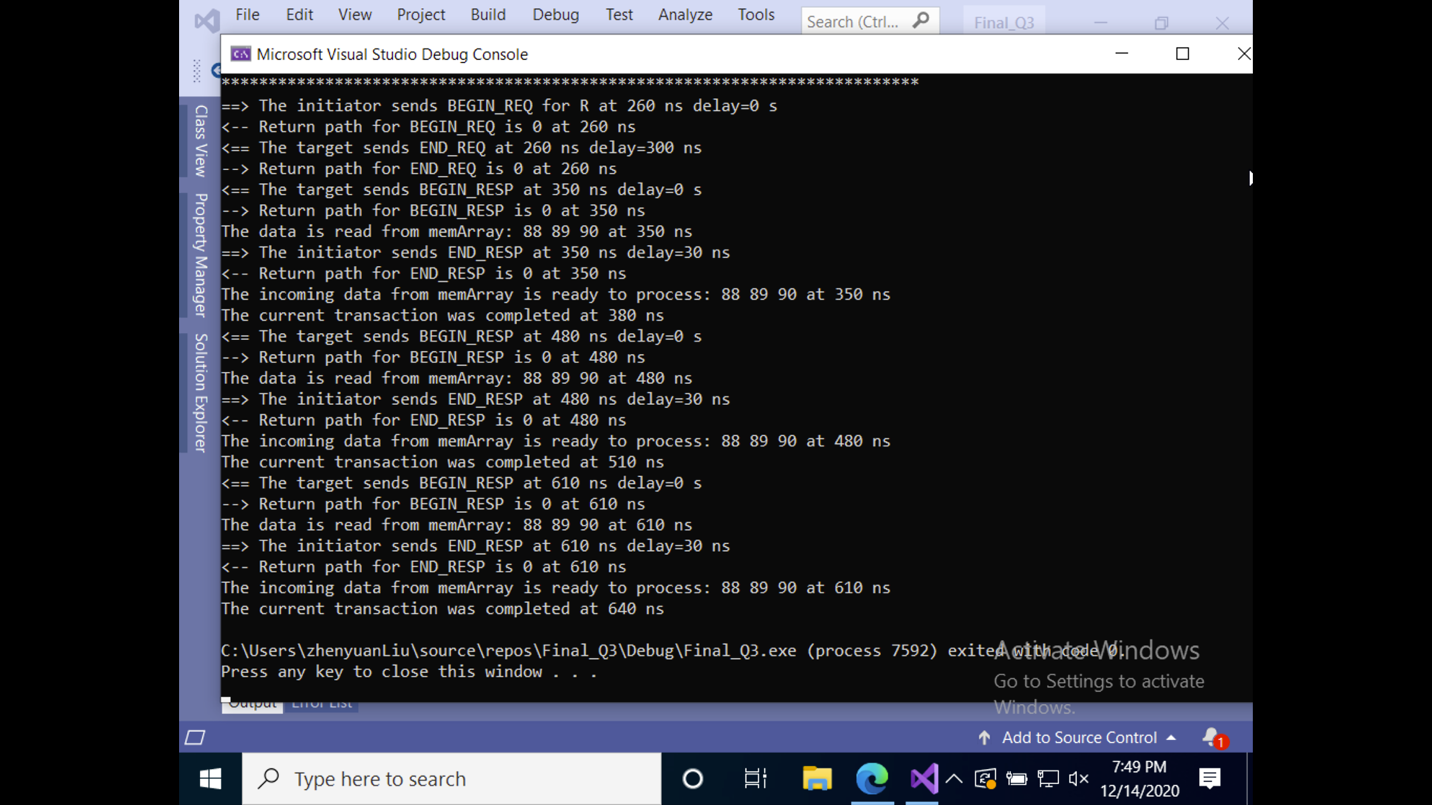


Figure 5. The first red box proved that the latency of target is 300ns. The second red box proved that the response accept delay is 30ns. The third red box proved that the request accept delay is 50ns. The code worked perfectly without the PEQ.