Lab-A dataflow debug and optimization

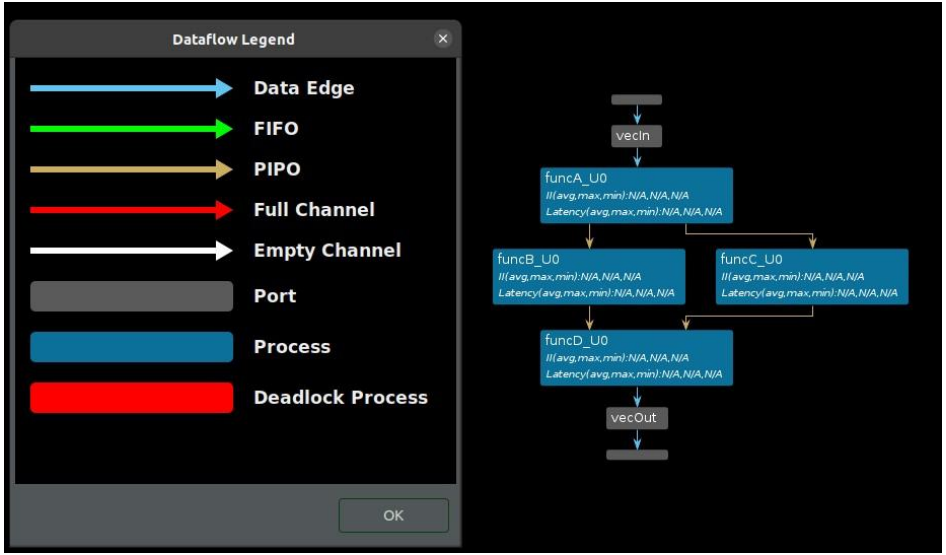
黃秉璿

LabA-Dataflow

- Dataflow viewer
- FIFO sizing for performance and avoiding deadlocks

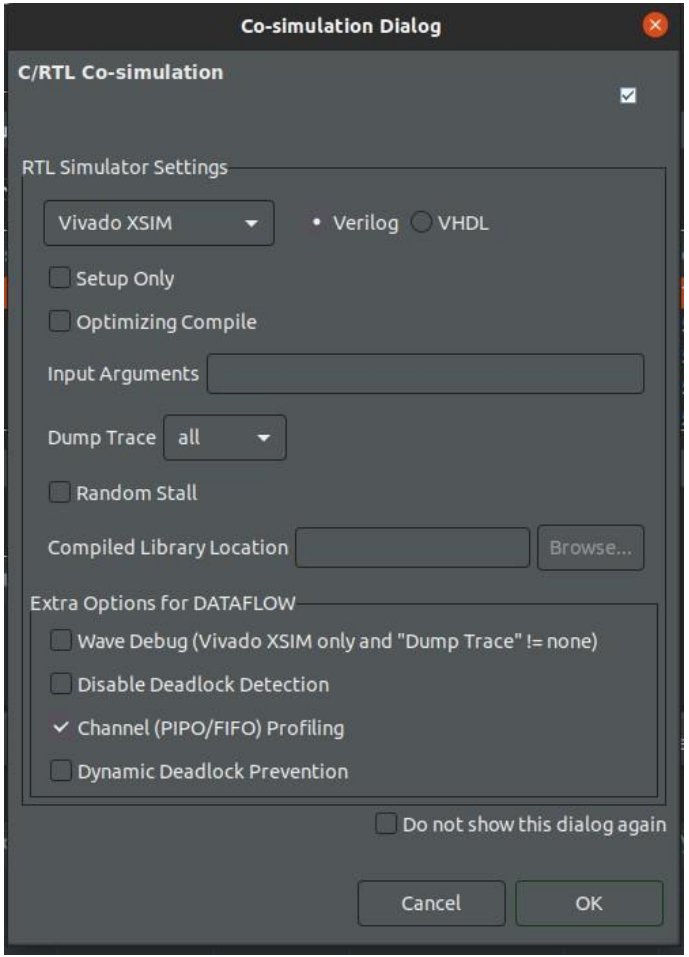
1. Dataflow Viewer

- After synthesis, we got dataflow viewer report
  - Process : function in the code



1. Dataflow Viewer

- In co-sim, select channel profiling to see the static properties of channel



1. Dataflow Viewer

- The process / channel profile includes II, process / FIFO statics

Console | Warnings | Guidance | Properties | Man Pages | Git Repositories | Modules/Loops | Errors | Dataflow

Process | Channel

Name	Cosim Category	FIFO EMPTY	FIFO FULL	Cosim Max Depth	Depth	Type	Sub-Type	BitWidth	Producer	Consumer	Cosim Distribution Graph
c1	N/A	N/A	N/A	N/A	0	PIPO	PIPO	8	funcA_U0	funcB_U0	N/A
c2	N/A	N/A	N/A	N/A	0	PIPO	PIPO	6	funcA_U0	funcC_U0	N/A
c3	N/A	N/A	N/A	N/A	0	PIPO	PIPO	8	funcB_U0	funcD_U0	N/A
c4	N/A	N/A	N/A	N/A	0	PIPO	PIPO	7	funcC_U0	funcD_U0	N/A

Process | Channel

Name	Cosim Category	Cosim Stalling Time	FIFO EMPTY	FIFO FULL	Cosim Stall No Start	Cosim Stall No Continue	Cosim AVG II	Cosim Max II	Cosim Min II	Cosim AVG Latency
funcA_U0	none	0.76%	0.00%	0.00%	0.00%	0.76%	50	51	50	51
funcB_U0	none	0.76%	0.00%	0.00%	0.38%	0.76%	51	51	51	52
funcC_U0	none	0.76%	0.00%	0.00%	0.38%	0.76%	51	51	51	52
funcD_U0	none	1.15%	0.00%	0.00%	1.15%	0.00%	52	53	51	50

1. Dataflow Viewer

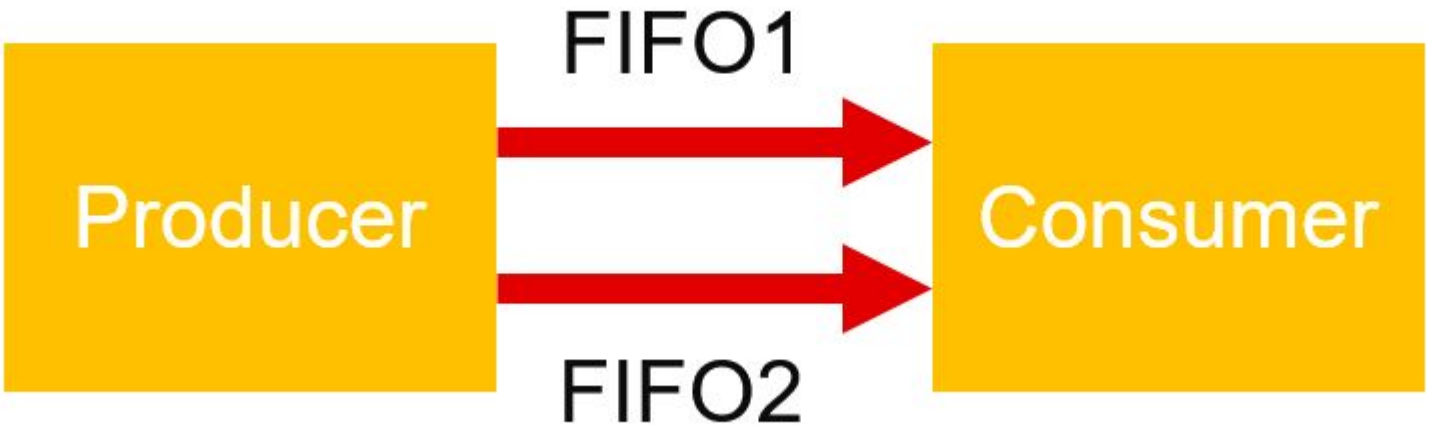
- Cosim Stalling Time : indicates what percentage of the simulation time was spent stalling for this particular process
- Cosim Read Block Time or Cosim Write Block Time shows the percentage of time blocked from reading or writing to the process's channels.
- Cosim Stall No Start and Cosim Stall No Continue indicates forward and back pressure respectively:
    - Forward pressure implies that you are prevented from starting another execution iteration by the block handshaking protocols
    - Back pressure implies that a consumer process is still processing the data that this process has produced and is not yet ready for the next set of data.

2. FIFO Sizing for Performance and avoiding Deadlocks

- **Types of channels**
  - **FIFO : sequential dataflow, explicitly controlled by scheduled read(), write(), empty() / full() will stall the process**
  - **PIPO (parallel in parallel out) : ping-pong buffer, controlled by ap_ready, ap_done signals**

2. FIFO Sizing for Performance and avoiding Deadlocks

- Insufficiently sized FIFOs (and PIPOs) in dataflow can cause deadlocks.
  - Case 1 flow: producer fifo1, fifo2, fifo1, … xN / consumer : fifo1, fifo2, fifo1, … xN
    - Required FIFO depth = 1
  - Case 2 flow: producer fifo1 x N, fifo2 xN / consumer : fifo1, fifo2, fifo1, … xN
    - Required FIFO1 depth = N



2. FIFO Sizing for Performance and avoiding Deadlocks

- Three ways for FIFO sizing
  - Manual FIFO sizing
    - In the process / channel profile -> modify depth
    - Back Annotate the New Depth into the Design
      - #pragma HLS stream depth=?? variable=??
  - Global FIFO sizing
    - Go to solution Settings -> General ->
    - config_dataflow -> override_user_fifo_depth
  - Automated FIFO sizing
    - Select Dynamic Deadlock Prevention when co-simulation