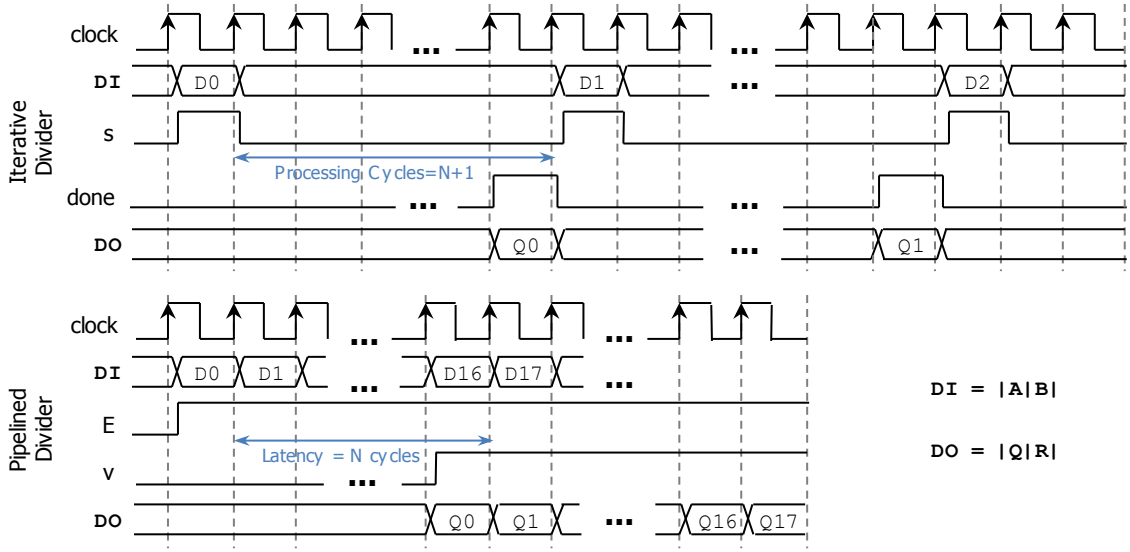
Solutions - Homework 3

(Due date: June 9th)

PROBLEM 1 (20 PTS)

- **Performance Analysis**: Iterative Integer Divider vs. Pipelined Integer Divider (N=M=16):
  - ✓ **Iterative Divider Operation**: Input data (16-bit A, 16-bit B) is read when the s signal (a one-cycle pulse) is asserted. After N+1=17 cycles, the result (16-bit Q, 16-bit R) is ready with done=1. Only after this, we can feed new data.
    To process data as fast as possible, we must issue s=1 (with new data) right after done=1.
  - ✓ **Pipelined Divider Operation**: The circuit reads input data (16-bit A, 16-bit B) when the enable (E) signal is asserted. After a processing delay of N=16 cycles, the result (16-bit Q, 16-bit R) is ready and it is signaled by v=1. Unlike the iterative divider, we can continuously feed data (with E=1).
    To process data as fast as possible, we must keep E=1 (with new data) every clock cycle.



- An operation is defined as the computation of one input data set. The processing cycles for P operations is given by:
  - ✓ Iterative Divider: It can compute P operations in P×(N+2) cycles (1 operation is processed in N+1 cycles, but there is a one cycle delay before we can start the next operation)
  - ✓ Pipelined Divider: It can compute P operations in N + (P-1) cycles.

- In the following table, complete the number of processing cycles, processing times (us), and operations per second.
  - ✓ Use $T_{CLOCK} = 8$ ns (same as the PL_CLK = 125 MHz input clock in ZYBO or ZYBO Z7-10)
  - ✓ The metric Operations per second is an average based on a given number of operations. Example: if a circuit can process 20 operations in 1 us, then we have $\frac{20\ operations}{1\ us} = 20 \times 10^6$ operations per second.

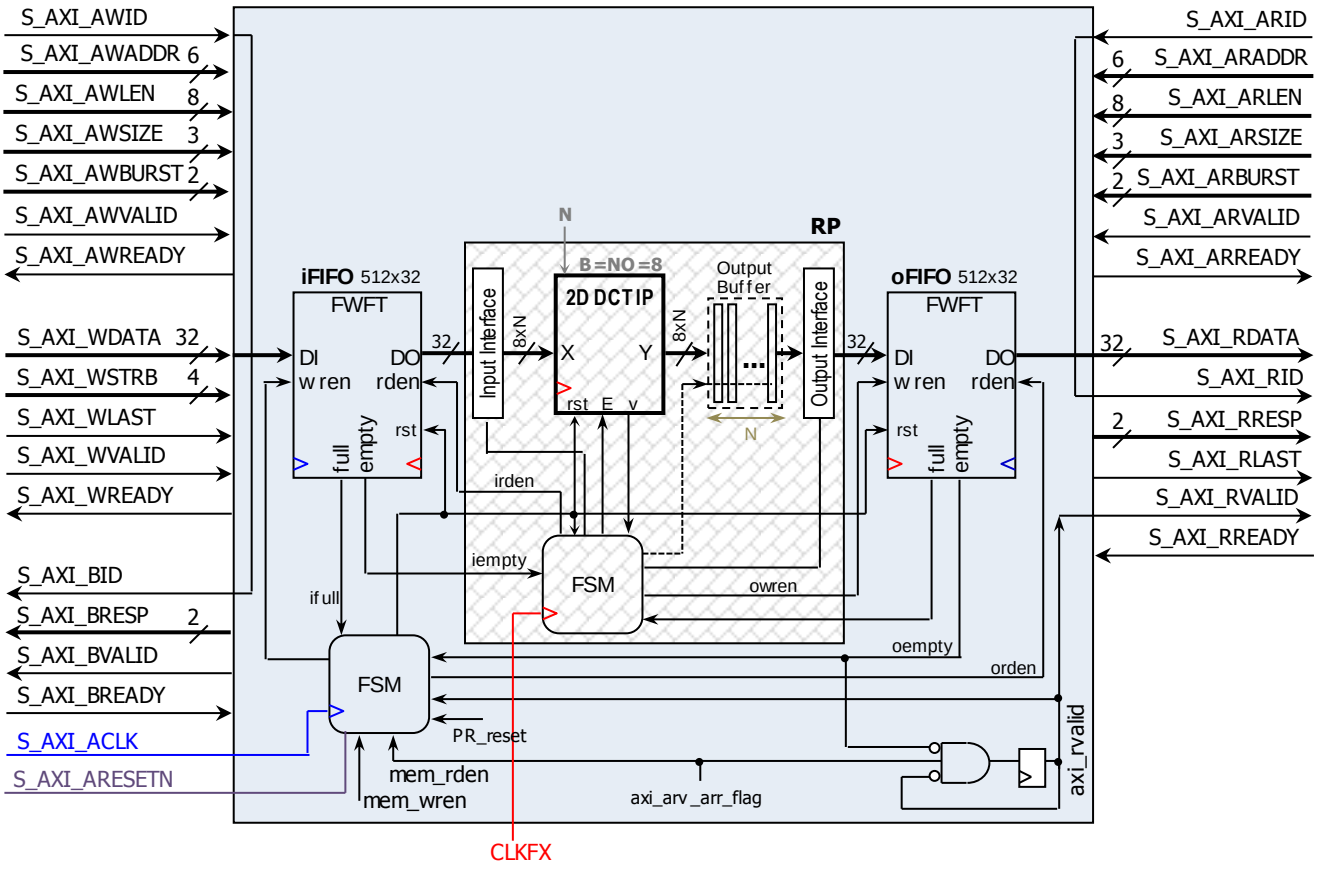
	Iterative Divider			Pipelined Divider		
P	Processing cycles	Processing Time (us)	Operations per second	Processing Cycles	Processing Time (us)	Operations per second
100	1800	14.4	6.944×10^6	115	0.92	1.087×10^8
1000	18000	144	6.944×10^6	1015	8.12	1.231×10^8
10000	180000	1440	6.944×10^6	10015	80.12	1.248×10^8
100000	1800000	14400	6.944×10^6	100015	800.12	1.249×10^8

- For the Iterative Divider: Is the Operations per second constant? Yes or No? Why?
  The operations per cycle is given by $\frac{P}{(N+2)P} = \frac{1}{N+2}$. This is a constant if N is constant.

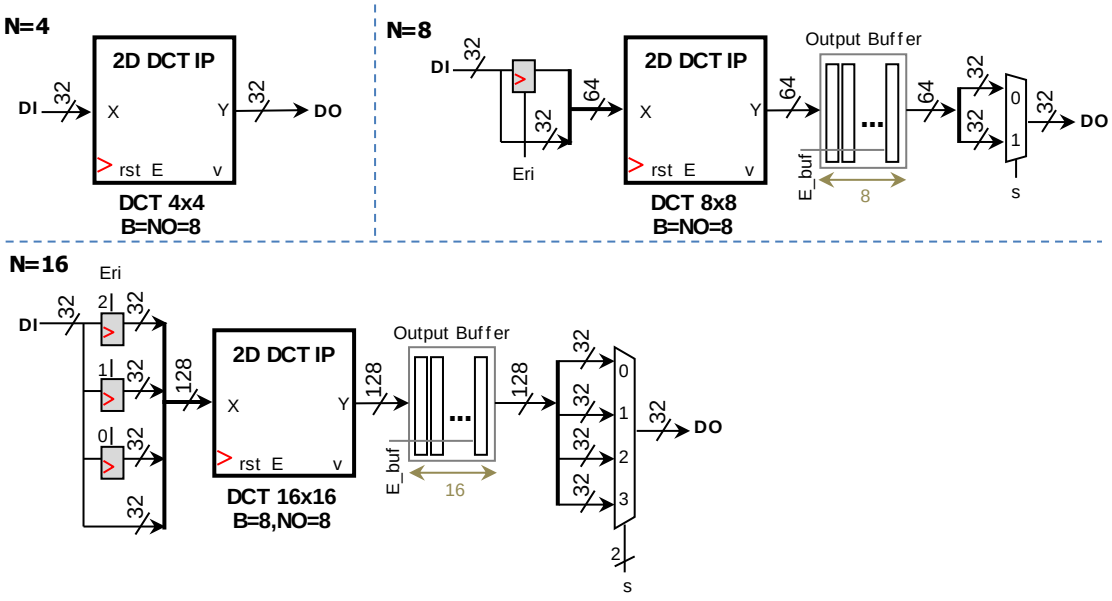
- For the Pipelined Divider: If P $\rightarrow\infty$:
  - ✓ How many operations are computed per cycle? $\frac{P}{N-1+P} = \frac{1}{\frac{N-1}{P}+1}\Big|_{P\rightarrow\infty}$ = 1 operation per cycle.
  - ✓ What is the Operations per second? 1 operation per cycle $\equiv \frac{1}{8\times10^{-9}} = 1.25 \times 10^8$ operations per second

PROBLEM 2 (15 PTS)

- The figure shows the 2D DCT IP AXI4-Full Peripheral. It includes a Reconfigurable Partition (RP). For this particular PR implementation, we allow for N to be run-time reconfigurable (N=4,8,16), while we fix the parameters B=NO=8.



- The input and output of the 2D DCT IP require more than 32 bits when N = 8, 16. This requires an Input interface to the iFIFO and an Output interface to the oFIFO. The figure shows the different interfaces for each N (4, 8, 16) when B=NO=8. As the FSM @ CLK_FX controls data flow from the input and the output, it depends on N.



- We want to build a dynamically reconfigurable system, where we can change N (4, 8, 16) at run-time:

  ✓ The RP (Reconfigurable Partition) is depicted in the figure. The Output Buffer, the Input interface and the output interface to FIFOs, as well as the FSM @ CLK_FX are included in the RP. Why is this necessary?

  The RP variations depend on N. When the parameter N changes, the input and output interfaces to FIFO change as well. The FSM @ CLK_FX is also dependent on N as it needs to know how much data is being transferred.

  ✓ Signal *rst*: Active-high signal generated by the FSM @ S_AXI_ACLK. It resets the 2D DCT IP, the red FSM, and the FIFOs. Why is this signal important? Do we assert this signal before or after performing DPR? Why?

  During DPR, the RP outputs toggle erratically and can write spurious data onto oFIFO. Also, the FFs of the RP are not cleared after DPR. So, we need to clear the RP FFs and the oFIFO. We do this via the rst signal.

  ✓ The RP outputs toggle during DPR. What could happen to the contents of oFIFO during DPR?

  Spurious data could be written onto oFIFO during DPR, as owren is an output of the RP.

PROBLEM 3 (65 PTS)

- Attach your Project Status Report (no more than 1 page, single-spaced, 2 columns, only one submission per group). This report should contain the current status of your project. For formatting, use the provided template (`Final Project - Report Template.docx`). The sections included in the template are the ones required in your Final Report. At this stage, you are only required to:
  - ✓ Include a project description.
  - ✓ Specify a (tentative) allocation of tasks in: i) software routine, and ii) reconfigurable hardware.
    - If you plan to use run-time alterable hardware, indicate what tasks it will be doing.
  - ✓ Hardware Architecture: Include a Draft Block Diagram with (tentative) I/O description and I/O mechanism.

- As a guideline, a generic hardware/software partitioning of an application is depicted. The figure shows the tasks performed by the software routine and the PS peripherals we plan to use. It also shows a Block Diagram of the Hardware with generic I/Os. The Reconfigurable Partition (RP) is also depicted. Note that this hardware uses external I/Os to the PL.