Pipelining for Performance Demo Script

Introduction

This demonstration script provides high-level instructions on using the PIPELINE directive.

Preparation:

- Required files: Necessary files are located at *C:\training\pipeline\demo*
- Required hardware: None
- Supporting materials: None

Pipelining for Performance

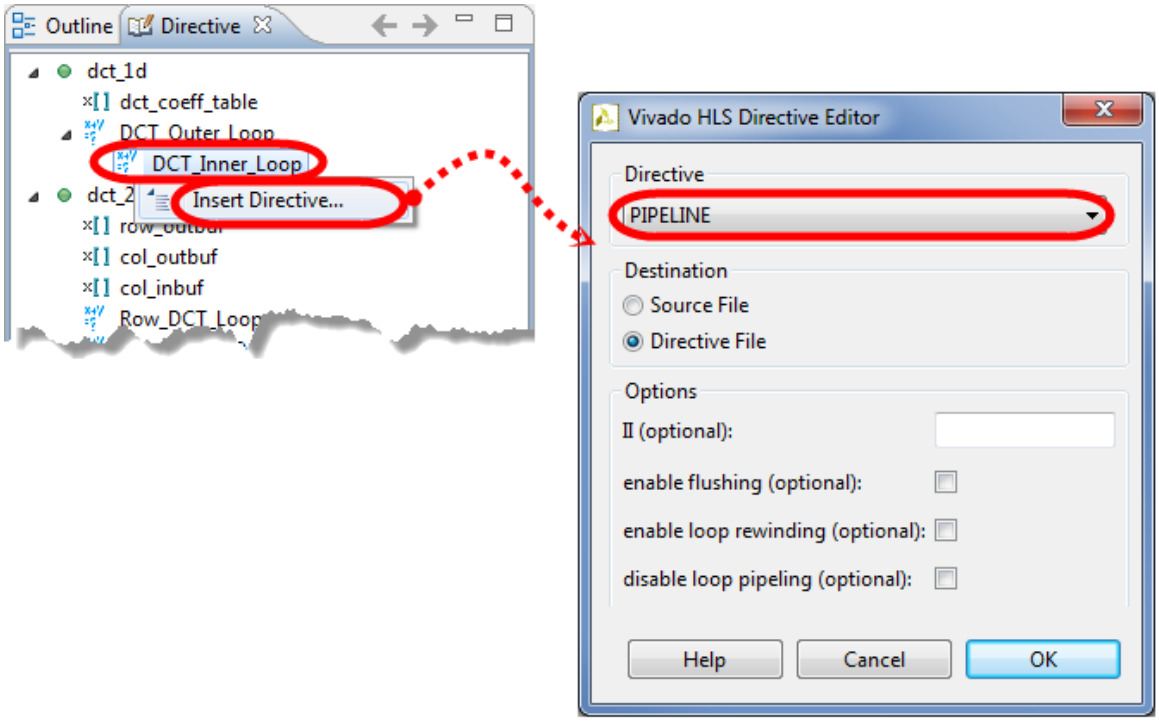
Action with Description	Point of Emphasis and Key Takeaway
• Launch the Vivado® HLS tool. • Open the provided **dct_prj** Vivado HLS tool project located at: *C:\training\pipeline\demo\dct_prj*	You can open existing Vivado HLS tool projects from the Vivado HLS tool Welcome page.

Action with Description	Point of Emphasis and Key Takeaway
• Access and review the source files (*dct.c* and *dct.h*) from the Explorer pane.	This C design uses a discrete cosine transformation (DCT). The function implements a 2D DCT algorithm by first processing each row of the input array via a 1D DCT, then processing the columns of the resulting array through the same 1D DCT. It calls the *read_data*, *dct_2d*, and *write_data* functions. The *read_data* function is defined at line 54 and consists of two loops: *RD_Loop_Row* and *RD_Loop_Col*. The *write_data* function is defined at line 66 and consists of two loops to perform writing the result. The *dct_2d* function, defined at line 23, calls the *dct_1d* function and performs transpose. Finally, the *dct_1d* function, defined at line 4, uses *dct_coeff_table* and performs the required function by implementing a basic iterative form of the 1D Type II DCT algorithm.

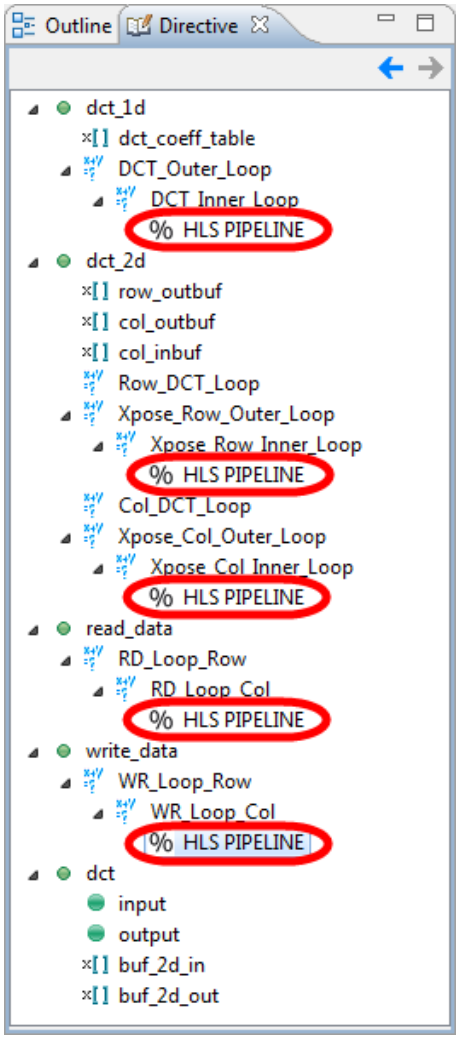
Action with Description	Point of Emphasis and Key Takeaway
• Run C synthesis. • Review the Synthesis report.	Once synthesis completes, the Synthesis report will open in the main viewing area. Notice that the estimated clock period is within the requested clock period. The Synthesis report also contains latency and throughput information of the design. The results correspond to the default solution (without any directives). You can further reduce the numbers down by specifying your requirements via directives. The Synthesis log is available in the Console pane.
What is the worst-case latency of the design? Answer: 2935	
In the Synthesis report, go to the Utilization Estimates section and note the number of DSP48E and block RAMs used to implement the solution. Number of BRAM_18K: 5 Number of DSP48E: 1	
• Double-click **dct_2d_csynth.rpt** to open the Synthesis report available under the *dct_prj* > *solution1* > *syn* > *report* folder in the Explorer pane.	What is the worst-case latency of the *dct_2d* function? Answer: 2644
• Similarly, open the **dct_1d2_csynth.rpt** file under the *dct_prj* > *solution1* > *syn* > *report* folder in the Explorer pane.	What is the worst-case latency of the *dct_1d* function? Answer: 145

Action with Description	Point of Emphasis and Key Takeaway
• Create a new solution named *solution2*. • Accept the default settings and click **Finish**. • Select **Project** > **Close Inactive Solution Tabs**.	Creating a new solution allows for different optimizations to be compared easily. There is no need to copy directives from the previous solution because the previous solution does not have any directives.
• Apply the **PIPELINE** directive on *DCT_Inner_Loop* of the *dct_1d* function (shown below).	Apply the PIPELINE directive on the inner loops. Leave the II field blank since the design tries to target II as 1; i.e., it will try to optimize the loop to accept a new input for every cycle. You will find the directive written to the *directive.tcl* file under the *dct_prj* > *solution2* > *constraints* folder. `set_directive_pipeline "dct_1d/DCT_Inner_Loop"`

Action with Description	Point of Emphasis and Key Takeaway

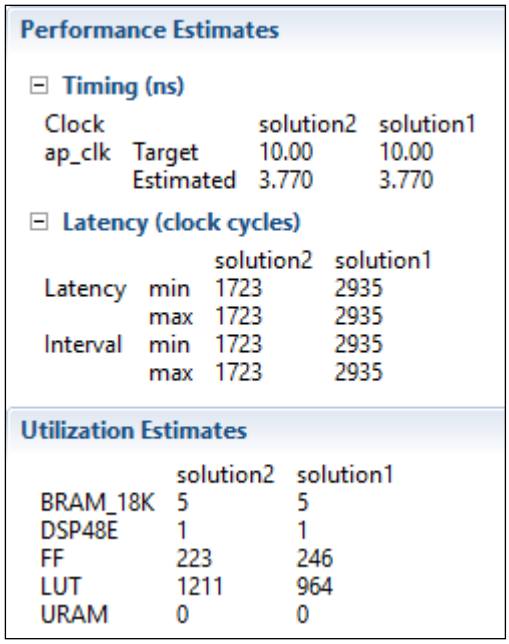


Action with Description	Point of Emphasis and Key Takeaway

• Similarly, apply the **PIPELINE** directive to the following loops:
  • *Xpose_Row_Inner_Loop* of the *dct_2d* function
  • *Xpose_Col_Inner_Loop* of the *dct_2d* function
  • *RD_Loop_Col* of the *read_data* function
  • *WR_Loop_Col* of the *write_data* function
	The Directive tab should look like the figure below after you finish applying the PIPELINE directive.
	 

• Run C synthesis.	Once synthesis completes, the Synthesis report will open in the main viewing area.

Action with Description	Point of Emphasis and Key Takeaway
• Compare the results of the two solutions (*solution1* and *solution2*). • Select **Project** > **Compare Reports**. Add the reports you wish to compare.	This allows you to compare the different optimizations of the project. You should see the comparison report as shown below.



Performance Estimates

Timing (ns)

Clock		solution2	solution1
ap_clk	Target	10.00	10.00
	Estimated	3.770	3.770

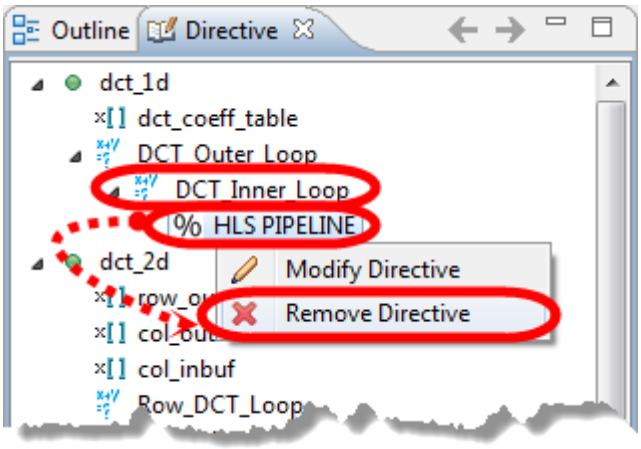
Latency (clock cycles)

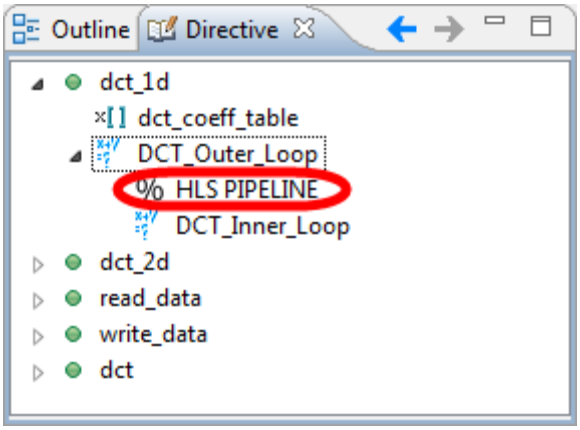
		solution2	solution1
Latency	min	1723	2935
	max	1723	2935
Interval	min	1723	2935
	max	1723	2935

Utilization Estimates

	solution2	solution1
BRAM_18K	5	5
DSP48E	1	1
FF	223	246
LUT	1211	964
URAM	0	0

• What is the worst-case latency of the design?	Answer: 1723
• Go to the Utilization Estimates section and note the number of DSP48E and block RAMs used to implement *solution2*.	Answer: Number of BRAM_18K: 5 Number of DSP48E: 1
• Double-click **dct_2d_csynth.rpt** to open the Synthesis report available under the *dct_prj* > *solution2* > *syn* > *report* folder in the Explorer pane.	What is the worst-case latency of the *dct_2d* function? Answer: 1590

Action with Description	Point of Emphasis and Key Takeaway
• Similarly, open the **dct_1d2_csynth.rpt** file under the *dct_prj* > *solution2* > *syn* > *report* folder in the Project Explorer pane.	What is the worst-case latency of the *dct_1d* function? Answer: 89
Note that the design latency and throughput (Interval) are improved with the application of the PIPELINE directive.	
• Create a new solution named *solution3*. • Accept the default settings and click **Finish**.	Moving the PIPELINE directive from the inner loop to the outer loop of *dct_1d* will lead to more parallelism of the multiply and add operations. That is, eight (8) multiply and add operations are performed concurrently, thus minimizing the number of cycles required to compute each value in the output array.
• Delete the **PIPELINE** directive from *DCT_Inner_Loop* of the *dct_1d* function.	Delete the inner loop pipeline and apply the PIPELINE directive from the outer loop.
	

<table>
<tr><th>Action with Description</th><th>Point of Emphasis and Key Takeaway</th></tr>
<tr><td>• Apply the PIPELINE directive on DCT_Outer_Loop of the dct_1d function.</td><td> 
</td></tr>
<tr><td>• Run C synthesis.
It is safe to ignore the warnings if any.</td><td>Once synthesis completes, the Synthesis report will open in the main viewing area.</td></tr>
<tr><td>• Compare the results of the two solutions (solution2 and solution3).</td><td>You should see the comparison report as shown below.</td></tr>
<tr><td colspan="2">
Performance Estimates
Timing (ns)
<table>
<tr><th>Clock</th><th></th><th>solution3</th><th>solution2</th></tr>
<tr><td>ap_clk</td><td>Target</td><td>10.00</td><td>10.00</td></tr>
<tr><td></td><td>Estimated</td><td>7.769</td><td>3.770</td></tr>
</table>
Latency (clock cycles)
<table>
<tr><th></th><th></th><th>solution3</th><th>solution2</th></tr>
<tr><td>Latency</td><td>min</td><td>843</td><td>1723</td></tr>
<tr><td></td><td>max</td><td>843</td><td>1723</td></tr>
<tr><td>Interval</td><td>min</td><td>843</td><td>1723</td></tr>
<tr><td></td><td>max</td><td>843</td><td>1723</td></tr>
</table>
Utilization Estimates
<table>
<tr><th></th><th>solution3</th><th>solution2</th></tr>
<tr><td>BRAM_18K</td><td>5</td><td>5</td></tr>
<tr><td>DSP48E</td><td>8</td><td>1</td></tr>
<tr><td>FF</td><td>491</td><td>223</td></tr>
<tr><td>LUT</td><td>1364</td><td>1211</td></tr>
<tr><td>URAM</td><td>0</td><td>0</td></tr>
</table>
</td></tr>
<tr><td colspan="2">Notice that throughput and latency are improved. With the PIPELINE directive applied on the outer loop, the inner loop automatically unrolls, which results in more resource utilization compared to solution2.</td></tr>
</table>

Summary

The PIPELINE directive inserts pipeline registers in the generated RTL to improve the latency of the design. When it was applied on the outer loop, it automatically unrolled the inner loops and inserted the required pipeline registers in the design. By applying this directive properly, you can increase the throughput of the design.

In this demo, you learned how to apply the PIPELINE directive and observed its impact on a design.

References:

- Supporting materials
  - *Vivado Design Suite Tutorial: High-Level Synthesis* (UG871)
  - *Vivado Design Suite User Guide: High-Level Synthesis* (UG902)