

Handling Memories Demo Script

Introduction

This demonstration script provides high-level instructions on how to remove bottlenecks caused by arrays in a C design.

Preparation:

- Required files: Necessary files are located at *C:\training\hls\demos\memory_optimization*
- Required hardware: None
- Supporting materials: None

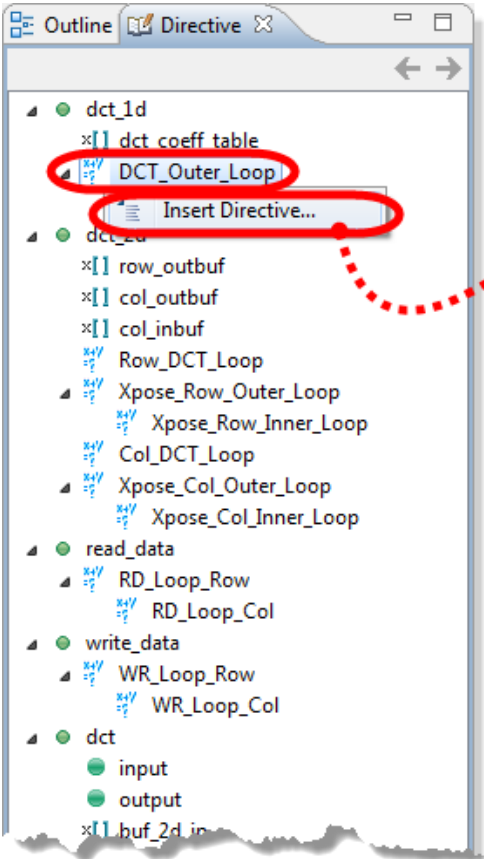
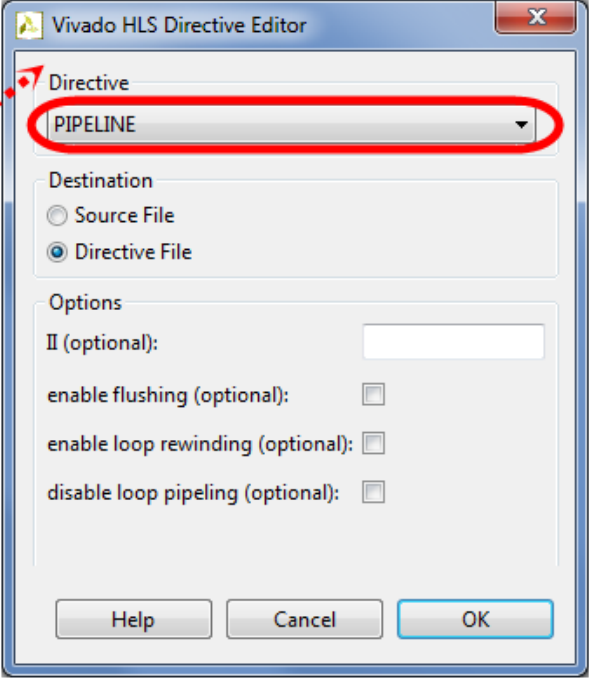
Handling Memories

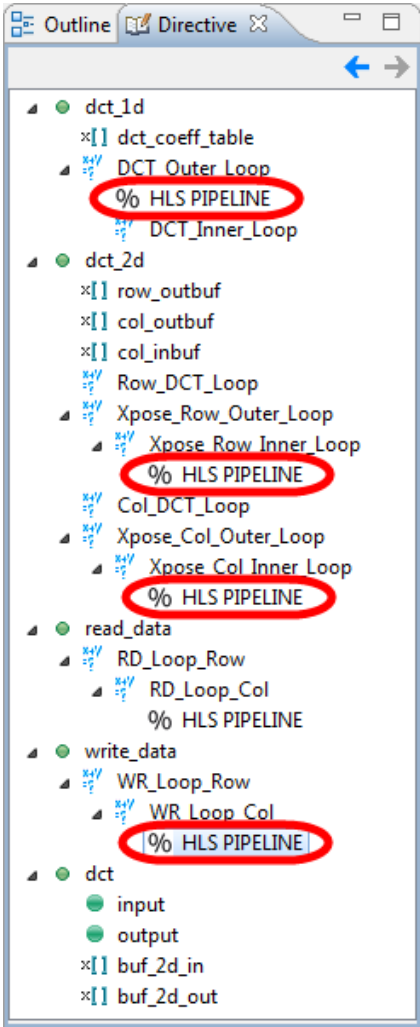
Action with Description	Point of Emphasis and Key Takeaway
<ul style="list-style-type: none">• Launch the Vivado® HLS tool.• Open the provided dct_prj Vivado HLS tool project located at: <i>C:\training\hls\demos\memory_optimization</i>	You can open existing Vivado HLS tool projects from the Vivado HLS tool Welcome page.

Action with Description	Point of Emphasis and Key Takeaway
<ul style="list-style-type: none"> Access and review the source files (<i>dct.c</i> and <i>dct.h</i>) from the Explorer pane. 	<p>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 <i>read_data</i>, <i>dct_2d</i>, and <i>write_data</i> functions.</p> <p>The <i>read_data</i> function is defined at line 54 and consists of two loops: RD_Loop_Row and RD_Loop_Col.</p> <p>The <i>write_data</i> function is defined at line 66 and consists of two loops to perform writing the result. The <i>dct_2d</i> function, defined at line 23, calls the <i>dct_1d</i> function and performs transpose.</p> <p>Finally, the <i>dct_1d</i> function, defined at line 4, uses <i>dct_coeff_table</i> and performs the required function by implementing a basic iterative form of the 1D Type II DCT algorithm.</p>

Action with Description	Point of Emphasis and Key Takeaway
<ul style="list-style-type: none"> Run C synthesis. Review the Synthesis report. 	<p>Once synthesis completes, the Synthesis report will open in the main viewing area.</p> <p>Notice that the estimated clock period is within the requested clock period.</p> <p>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.</p> <p>The Synthesis log is available in the Console pane.</p>
<p>What is the worst-case latency of the design?</p> <p>Answer: 6647</p>	
<ul style="list-style-type: none"> Create a new solution named <i>solution2</i>. Accept the default settings and click Finish. 	<p>Creating a new solution allows for different optimizations to be compared easily.</p> <p>There is no need to copy directives from the previous solution because the previous solution does not have any directives. Even if the directives are copied (the default setting), there would be no impact to the demo since there are no directives in the initial solution.</p>
<p>As part of the optimization process, as seen in the "Pipeline for Performance" demo, you will begin by pipelining the loops in the design.</p>	

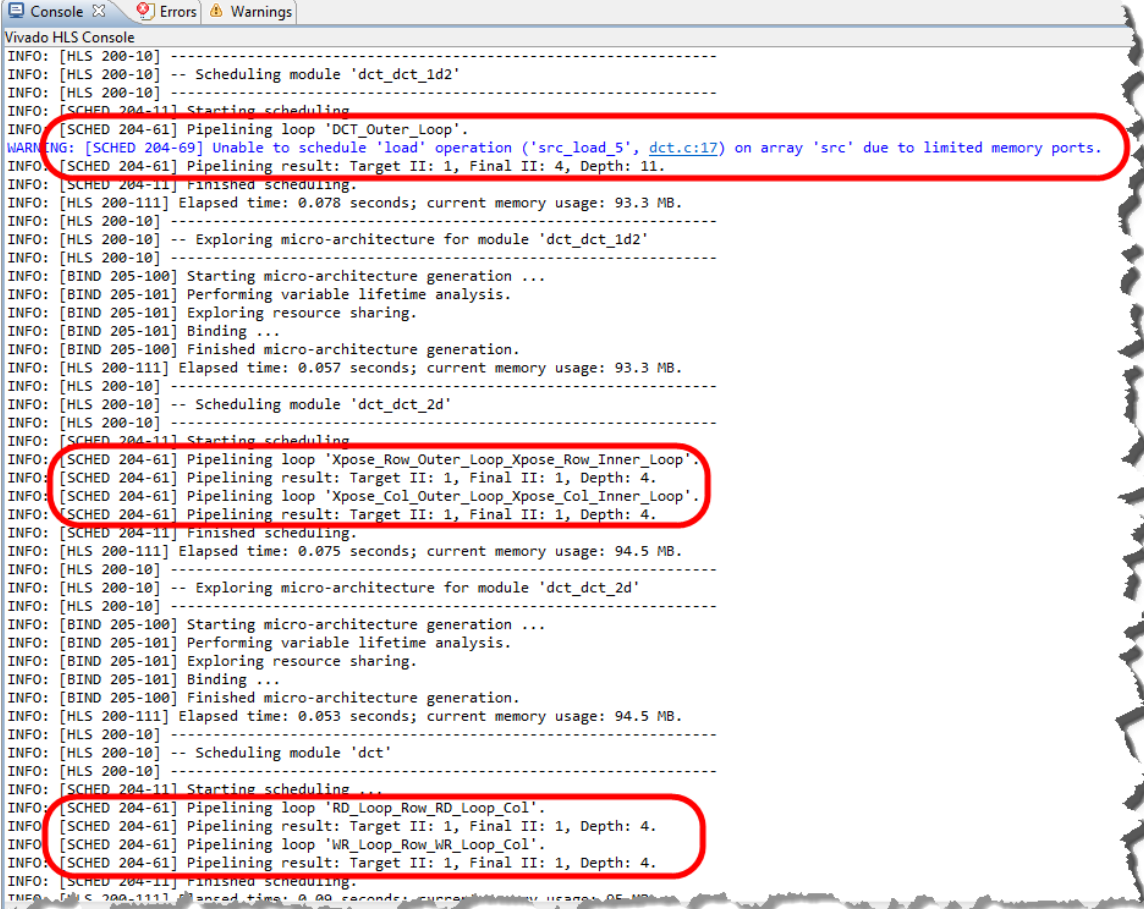
Action with Description	Point of Emphasis and Key Takeaway
<ul style="list-style-type: none"> Apply the PIPELINE directive on <i>DCT_Outer_Loop</i> of the <i>dct_1d</i> function (shown below). 	<p>Apply the PIPELINE directive on the outer loop.</p> <p>Moving the PIPELINE directive from the inner loop to the outer loop of <i>dct_1d</i> will lead to more parallelism of the multiply and add operations.</p> <p>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.</p> <p>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.</p> <p>You will find the directive written to the <i>directive.tcl</i> file under the <i>dct_prj > solution2 > constraints</i> folder.</p> <pre>set_directive_pipeline "dct_1d/DCT_Outer_Loop"</pre>

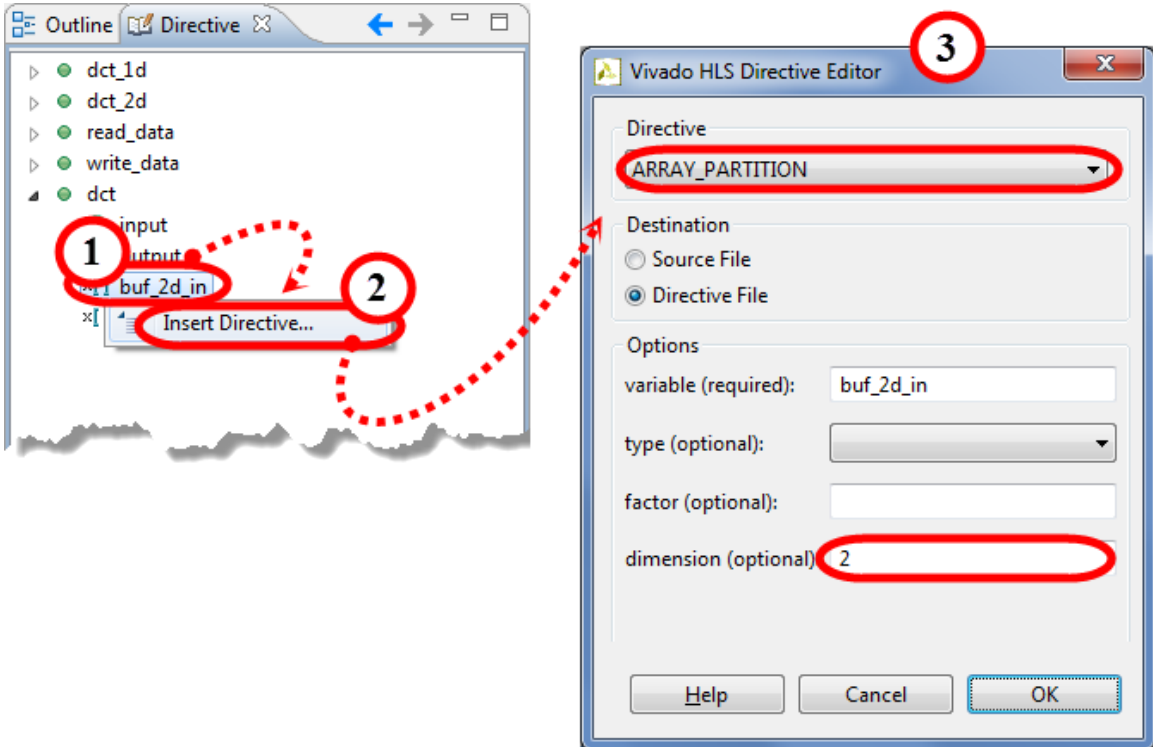
Action with Description	Point of Emphasis and Key Takeaway
	

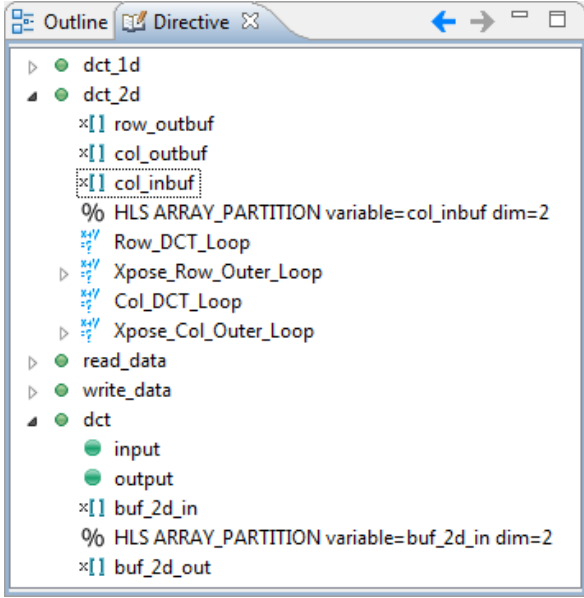
Action with Description	Point of Emphasis and Key Takeaway
<ul style="list-style-type: none"> Similarly, apply the PIPELINE directive to the following loops: <ul style="list-style-type: none"> <i>Xpose_Row_Inner_Loop</i> of the <i>dct_2d</i> function <i>Xpose_Col_Inner_Loop</i> of the <i>dct_2d</i> function <i>RD_Loop_Col</i> of the <i>read_data</i> function <i>WR_Loop_Col</i> of the <i>write_data</i> function 	<p>The Directive tab should look like the figure below after you finish applying the PIPELINE directive.</p>  <p>The screenshot shows the 'Directive' tab in a software interface. The project tree is expanded, showing the following structure:</p> <ul style="list-style-type: none"> dct_1d <ul style="list-style-type: none"> dct_coeff_table DCT_Outer_Loop (highlighted with a red circle and labeled '% HLS PIPELINE') DCT_Inner_Loop dct_2d <ul style="list-style-type: none"> row_outbuf col_outbuf col_inbuf Row_DCT_Loop Xpose_Row_Outer_Loop <ul style="list-style-type: none"> Xpose_Row_Inner_Loop (highlighted with a red circle and labeled '% HLS PIPELINE') Col_DCT_Loop Xpose_Col_Outer_Loop <ul style="list-style-type: none"> Xpose_Col_Inner_Loop (highlighted with a red circle and labeled '% HLS PIPELINE') read_data <ul style="list-style-type: none"> RD_Loop_Row <ul style="list-style-type: none"> RD_Loop_Col (highlighted with a red circle and labeled '% HLS PIPELINE') write_data <ul style="list-style-type: none"> WR_Loop_Row <ul style="list-style-type: none"> WR_Loop_Col (highlighted with a red circle and labeled '% HLS PIPELINE') dct <ul style="list-style-type: none"> input output buf_2d_in buf_2d_out
<ul style="list-style-type: none"> Run C synthesis. 	<p>Once synthesis completes, the Synthesis report will open in the main viewing area.</p>
<ul style="list-style-type: none"> Compare the results of two solutions (<i>solution1</i> and <i>solution2</i>). 	<p>This allows you to compare the different optimizations of the project.</p> <p>You should see the comparison report as shown below.</p>

Action with Description	Point of Emphasis and Key Takeaway																																															
<div><div>Performance Estimates</div><div><div>Timing (ns)</div><table><tr><td>Clock</td><td></td><td>solution2</td><td>solution1</td></tr><tr><td>ap_clk</td><td>Target</td><td>4.00</td><td>4.00</td></tr><tr><td></td><td>Estimated</td><td>3.48</td><td>3.48</td></tr></table></div><div><div>Latency (clock cycles)</div><table><tr><td></td><td></td><td>solution2</td><td>solution1</td></tr><tr><td>Latency</td><td>min</td><td>946</td><td>6647</td></tr><tr><td></td><td>max</td><td>946</td><td>6647</td></tr><tr><td>Interval</td><td>min</td><td>947</td><td>6648</td></tr><tr><td></td><td>max</td><td>947</td><td>6648</td></tr></table></div><div><div>Utilization Estimates</div><table><tr><td></td><td>solution2</td><td>solution1</td></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>849</td><td>384</td></tr><tr><td>LUT</td><td>556</td><td>362</td></tr></table></div></div>		Clock		solution2	solution1	ap_clk	Target	4.00	4.00		Estimated	3.48	3.48			solution2	solution1	Latency	min	946	6647		max	946	6647	Interval	min	947	6648		max	947	6648		solution2	solution1	BRAM_18K	5	5	DSP48E	8	1	FF	849	384	LUT	556	362
Clock		solution2	solution1																																													
ap_clk	Target	4.00	4.00																																													
	Estimated	3.48	3.48																																													
		solution2	solution1																																													
Latency	min	946	6647																																													
	max	946	6647																																													
Interval	min	947	6648																																													
	max	947	6648																																													
	solution2	solution1																																														
BRAM_18K	5	5																																														
DSP48E	8	1																																														
FF	849	384																																														
LUT	556	362																																														
<ul style="list-style-type: none">What is the worst-case latency of the design?	Answer: 946																																															
<ul style="list-style-type: none">Go to the Utilization Estimates section and note the number of DSP48E and block RAMs used to implement <i>solution2</i>.	Answer: Number of BRAM_18K: 5 Number of DSP48E: 8																																															

Action with Description	Point of Emphasis and Key Takeaway
<ul style="list-style-type: none"> Select the Console tab and review the synthesis information. 	<p>From the Synthesis log, note that the design was not able to achieve the requested II on <i>DCT_Outer_Loop</i> because of the limited memory ports on the <i>src</i> element.</p> <p>The <i>src</i> is input to the <i>dct_1d</i> function and <i>dct_1d</i> is called twice in the <i>dct_2d</i> function (line no. 33 & line no. 44).</p> <p>At line 33, <i>in_block</i> is accessed via the <i>src</i> element in <i>dct_1d</i>. At line no. 44, <i>col_inbuf</i> is accessed via the <i>src</i> element in <i>dct_1d</i>.</p> <p>Therefore, you will need to partition both the <i>col_inbuf</i> and <i>in_block</i> arrays to achieve a throughput of 1.</p>

Action with Description	Point of Emphasis and Key Takeaway
	
<p>You will now solve the <i>dct_1d</i> pipeline II problem by increasing the memory bandwidth available to it.</p> <p>This will be done by partitioning the arrays from which the <i>dct_1d</i> inner loops read data (<i>in_block</i> in <i>Row_DCT_Loop</i> and <i>col_inbuf</i> in <i>Col_DCT_Loop</i>).</p>	
<ul style="list-style-type: none"> Create a new solution named <i>solution3</i>. Accept the default settings and click Finish. 	<p>In this solution, you will apply the <code>ARRAY_PARTITION</code> directive to <i>buf_2d_in</i> of the <i>dct</i> function and <i>col_inbuf</i> of the <i>dct2d</i> function.</p>

Action with Description	Point of Emphasis and Key Takeaway
<ul style="list-style-type: none">Apply the ARRAY_PARTITION directive to <i>buf_2d_in</i> of the <i>dct</i> function.	Partitioning large arrays into multiple smaller arrays or into individual registers can help improve access to data and remove block RAM bottlenecks.
 <p>The screenshot illustrates the process of applying the ARRAY_PARTITION directive in Vivado HLS. On the left, the 'Outline' pane shows a project structure with a function 'dct' containing an array 'buf_2d_in'. A red circle labeled '1' highlights 'buf_2d_in', and a red circle labeled '2' highlights the 'Insert Directive...' button. A red dotted arrow points from the 'Insert Directive...' button to the 'Vivado HLS Directive Editor' dialog box on the right. In the dialog box, a red circle labeled '3' highlights the 'Directive' dropdown menu, which is set to 'ARRAY_PARTITION'. Below this, the 'Options' section shows 'variable (required):' set to 'buf_2d_in' and 'dimension (optional):' set to '2'. The 'Destination' section has 'Directive File' selected. At the bottom are 'Help', 'Cancel', and 'OK' buttons.</p>	

Action with Description	Point of Emphasis and Key Takeaway
<ul style="list-style-type: none"> Similarly, apply the ARRAY_PARTITION directive <i>col_inbuf</i> of the <i>dct_2d</i> function. 	<p>The Directive tab should look like the figure below after you finish applying the ARRAY_PARTITION directive.</p> 
<ul style="list-style-type: none"> Run C synthesis. 	<p>Once synthesis completes, the synthesis report will open in the main viewing area.</p>
<p>Examine the Synthesis log. Has the PIPELINE II directive been met?</p> <p>Yes, the pipeline directive met II=1. The memory bandwidth increased via the array partitioning and thus the design met the requested II value.</p>	
<ul style="list-style-type: none"> Compare the results of the two solutions (<i>solution2</i> and <i>solution3</i>). 	<p>You should see the comparison report as shown below.</p>

Action with Description	Point of Emphasis and Key Takeaway																																																
<div><div>Performance Estimates</div><div><div>Timing (ns)</div><table><tr><td>Clock</td><td></td><td>solution3</td><td>solution2</td></tr><tr><td>ap_clk</td><td>Target</td><td>4.00</td><td>4.00</td></tr><tr><td></td><td>Estimated</td><td>3.48</td><td>3.48</td></tr></table></div><div><div>Latency (clock cycles)</div><table><tr><td></td><td></td><td>solution3</td><td>solution2</td></tr><tr><td>Latency</td><td>min</td><td>548</td><td>946</td></tr><tr><td></td><td>max</td><td>548</td><td>946</td></tr><tr><td>Interval</td><td>min</td><td>549</td><td>947</td></tr><tr><td></td><td>max</td><td>549</td><td>947</td></tr></table></div><div><div>Utilization Estimates</div><table><tr><td></td><td>solution3</td><td>solution2</td></tr><tr><td>BRAM_18K</td><td>3</td><td>5</td></tr><tr><td>DSP48E</td><td>8</td><td>8</td></tr><tr><td>FF</td><td>1247</td><td>849</td></tr><tr><td>LUT</td><td>653</td><td>556</td></tr></table></div></div>		Clock		solution3	solution2	ap_clk	Target	4.00	4.00		Estimated	3.48	3.48			solution3	solution2	Latency	min	548	946		max	548	946	Interval	min	549	947		max	549	947		solution3	solution2	BRAM_18K	3	5	DSP48E	8	8	FF	1247	849	LUT	653	556	
Clock		solution3	solution2																																														
ap_clk	Target	4.00	4.00																																														
	Estimated	3.48	3.48																																														
		solution3	solution2																																														
Latency	min	548	946																																														
	max	548	946																																														
Interval	min	549	947																																														
	max	549	947																																														
	solution3	solution2																																															
BRAM_18K	3	5																																															
DSP48E	8	8																																															
FF	1247	849																																															
LUT	653	556																																															
<p>Latency was reduced to 548. Block RAM usage decreased to 3 from 5.</p> <p>Memory utilization will usually be more after array partitioning. But in this case, some of the memory elements were implemented in the distributed RAMs/ROMS (you can observe this in the Synthesis log in the Console tab) and hence the number of block RAMs was actually reduced compared to the previous solution.</p>																																																	

Summary

Memory elements may become bottlenecks when it comes to meeting throughput and latency requirements. In this demo, you learned how to apply a partition directive on arrays in the design and observed its impact on resources and throughput.

References:

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