Project1: FIR Filter Design

Cheng-Yu Chen

Architectures:

Optimized1: Coefficient array and delay_line array are set to be signed char array.

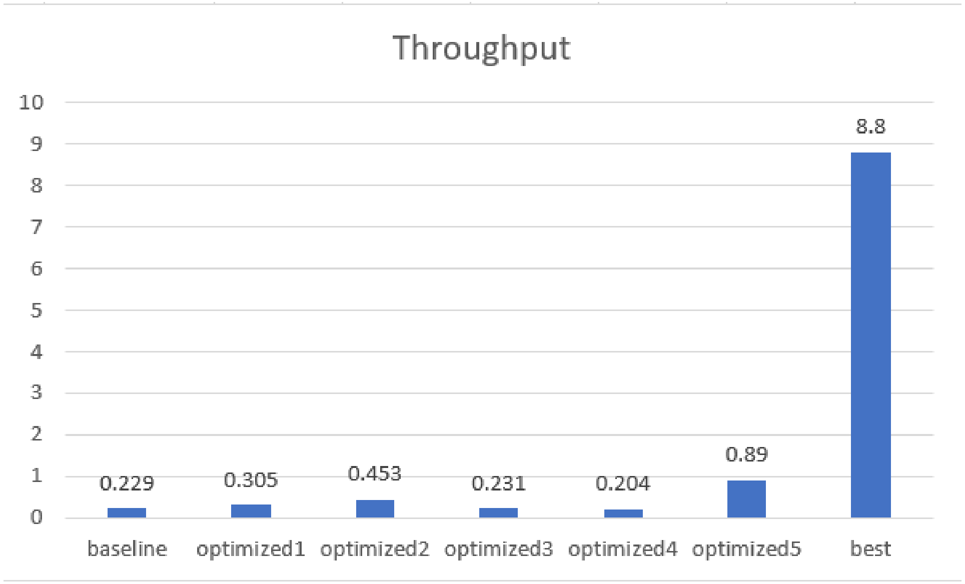
Optimized2: It is set to be pipelining and II is set to be 1.

Optimized3: Conditional statement is removed so i=0 is executed outside the loop.

Optimized4: The loop is divided into two parts; One is from i=N-1 to i=(N/2)+1 and the other is from i=N/2 to i=0.

Optimized5: The delay_line is set to be registers.

best: It combined the feature of optimized3, optimized5, and loop unrolling.



Question 1 - Variable Bitwidths:

Change the bitwidth of the variables inside the function body (do not change the bitwidth of the parameters). How does the bitwidth affect the performance?

Throughput: baseline=229KHz, optimized1=305KHz

When the bitwidth is smaller, the throughput is higher.

How does it affect the resource usage?

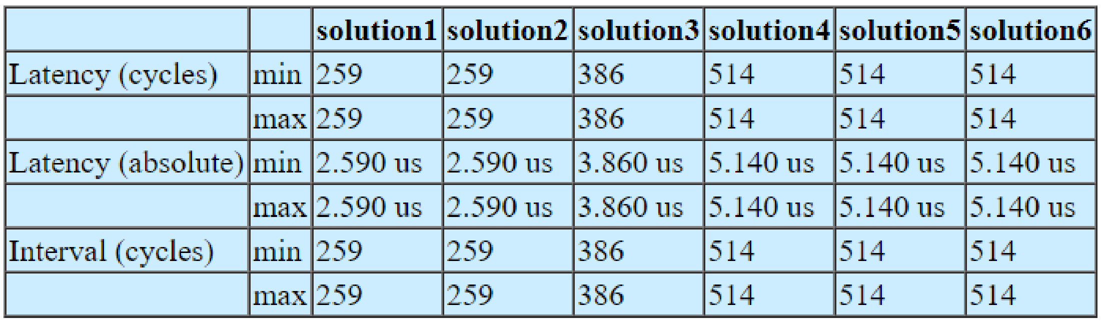
When the bitwidth is smaller, expression, multiplexer, and registers all have less total bits compared with baseline.

What is the minimum data size that you can use without losing accuracy (i.e., your results still match the golden output)?

8bit; it's because all elements of delay_line and coefficient array are less than 128.

Question 2 - Pipelining:

Explicitly set the loop initiation interval (II) starting at 1 and increasing in increments of 1 cycle. How does increasing the II affect the loop latency? What are the trends?



		solution1	solution2	solution3	solution4	solution5	solution6
Latency (cycles)	min	259	259	386	514	514	514
	max	259	259	386	514	514	514
Latency (absolute)	min	2.590 us	2.590 us	3.860 us	5.140 us	5.140 us	5.140 us
	max	2.590 us	2.590 us	3.860 us	5.140 us	5.140 us	5.140 us
Interval (cycles)	min	259	259	386	514	514	514
	max	259	259	386	514	514	514

Solution1 is II=1, Solution2 is II=2, and Solution3 is II=3, Solution1 is II=4, Solution2 is II=5, and Solution3 is II=6

Increasing II will increase the latency; However, it stop increasing after II is four.

At some point setting the II to a larger value does not make sense. What is that value in this example? How would you calculate that value for a general for loop?

In this example, it's 4; II means number of clock cycles before the function can accept new input data so when II is over iteration latency of the loop, the setting is useless.

Question 3 - Removing Conditional Statements:

Rewrite the code to remove any conditional statements. Compare the designs with and without if/else condition. Is there a difference in performance and/or resource utilization?

Throughput: baseline=229KHz, optimized3=231KHz

Without if/else condition, optimized3 has less LUT and FF. Also, it has higher throughput.

Does the presence of the conditional branch have any effect when the design is pipelined? If so, how and why?

Throughput: optimized2=457KHz, optimized3(with pipeline)=897KHz

Without the conditional branch, optimized3(with pipeline) has almost double throughput. The reason is that it does less work in every loop iteration so the latency of every loop iteration is smaller. Therefore, it could achieve lower initiation interval and have higher throughput.

Question 4 - Loop Partitioning:

Is there an opportunity for loop partitioning in FIR filters?

No, it's because two loops have some data dependency.

Compare your hardware designs before and after loop partitioning. What is the difference in performance?

Throughput: baseline=229KHz, optimized4=204KHz

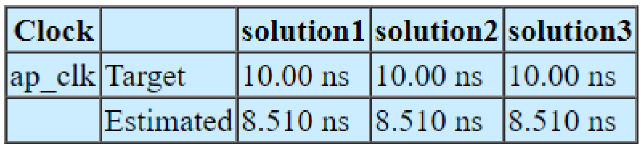
Optimized4 have lower throughput since it can't execute loops parallelly.

How do the number of resources change? Why?

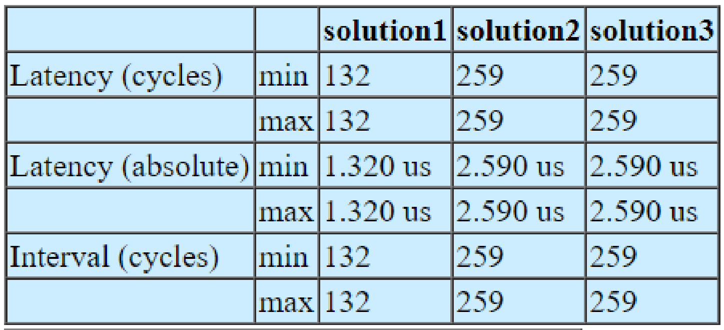
Optimized4 have more DSP48E, FF, and LUT since it have two loops and each loop have its own resources.

Question 5 - Memory Partitioning:

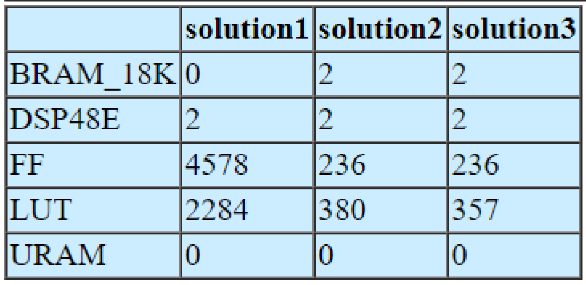
Compare the memory partitioning parameters: block, cyclic, and complete. What is the difference in performance and resource usage (particularly with respect to BRAMs and FFs)?



Clock		solution1	solution2	solution3
ap_clk	Target	10.00 ns	10.00 ns	10.00 ns
	Estimated	8.510 ns	8.510 ns	8.510 ns



		solution1	solution2	solution3
Latency (cycles)	min	132	259	259
	max	132	259	259
Latency (absolute)	min	1.320 us	2.590 us	2.590 us
	max	1.320 us	2.590 us	2.590 us
Interval (cycles)	min	132	259	259
	max	132	259	259



	solution1	solution2	solution3
BRAM_18K	0	2	2
DSP48E	2	2	2
FF	4578	236	236
LUT	2284	380	357
URAM	0	0	0

Solution1 is complete, Solution2 is block(factor=2), and Solution3 is cyclic(factor=2).
For optimized5, I use pipeline and memory partition.
Throughput: complete=890KHz, block=454KHz, cyclic=454KHz
Complete one has higher throughput. Cyclic and block one have same throughput.
Complete one has much more FF but doesn't have BRAM.
Block and cyclic one have two BRAM and have same numbers of FF.
Which one gives the best performance? Why?
Complete one has the best performance since all elements of its delay_line are registers and registers take less time to get data. Therefore, the latency of the pipeline could be smaller.

Question 6 - Best Design:

Combine any number of optimizations to get your best architecture. In what way is it the best?
I combine everything beneficial to throughput so my design is best in throughput.
What optimizations did you use to obtain this result?

I used loop unrolling since it’s helpful to process data parallelly. Also, I remove conditional statement to reduce the delay of multiplexers. Plus, I used registers for arrays so the data could be read faster..