Winter 2021

493-0: VLSI Algorithmics

Lab 4. Working with Memories

Disclaimer: This tutorial was based on original ***Mentor Graphics*** tutorials. Slight modifications were required due to divergences in the versions of the tools and deprecated instructions. All credits go to the authors of the original tutorials.

In this lab you will learn how to:

- (1) Map C++ arrays to memories
- (2) Analyze and debug pipeline failures due to memory resource conflicts
- (3) Use memory word-width, interleave, and block_size constraints to improve performance

---

Before we begin with the tutorial...

Remember to check Tutorial 0 to assert how to setup ssh to access the Wilkinson server and configure and run Catapult for the first time. The link is here.

Part 1 - Steps:

0. Access the Wilkinson server with your NetID and open a Terminal.

1. Load the cad tools environment by running command **(1)**:

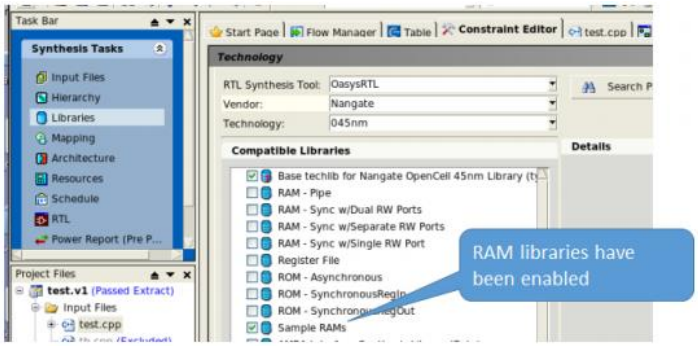
```
>> source ~/envsetup-catapult.sh
```
(1)

2. Navigate to the Labs directory and then to *Lab4* using command **(2)**:

```
>> cd ~/Labs/Lab4
```
(2)

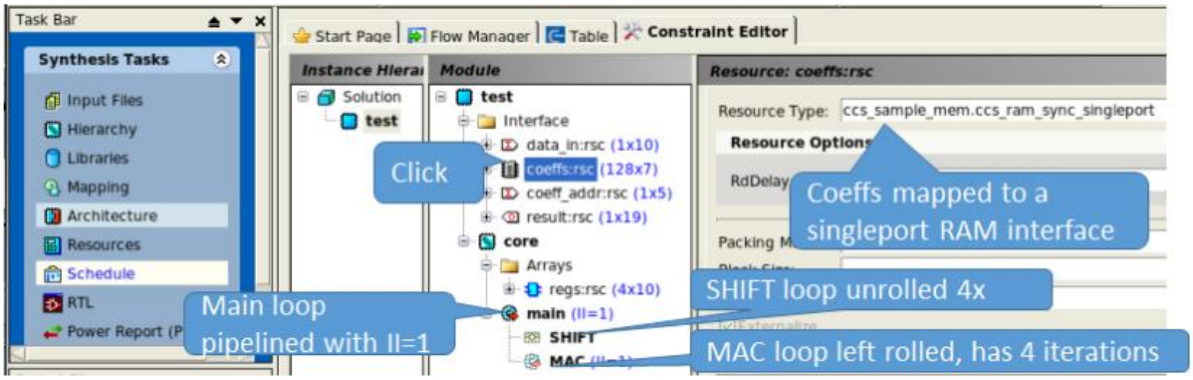
3. Launch Catapult by typing “catapult -product ultra” at the terminal prompt
4. Go to File > Run Script and select “directives_mem_merge.tcl” file and click open. This will load input files and compile and synthesize a design that uses Catapult memory libraries to map arrays to memories.
5. In Project Files > Input Files folder double-click on test.cpp. Note the following:
    a. Data “data_in” is streamed from an ac_channel a shifted into 4-element shift register
    b. “coeffs” is an array on the design interface with 32x4 elements.
    c. The MAC loop has 4 iterations and multiplies the shift register against 4 values from coeffs selected by “addr”

6. Click on Libraries in the Task Bar and note that the RAM libraries have been selected.

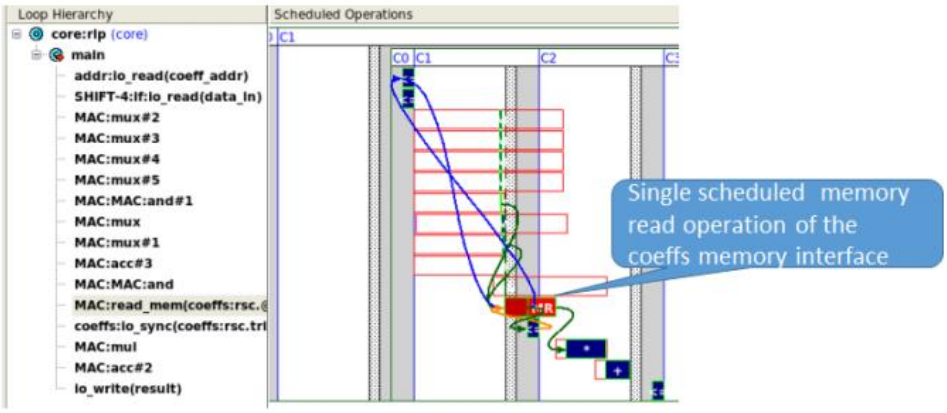


7. Click on Architecture in the Task Bar and note:
    a. The coeffs array has been mapped to a singleport RAM interface
    b. The main loop is pipelined with II=1.

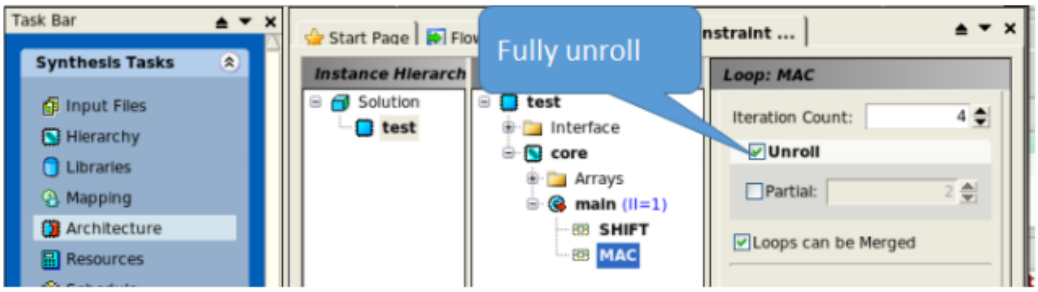
c. The SHIFT loop is fully unrolled
d. The MAC loop is left rolled and has 4 iterations.



8. Look at the Table View and note that the design has a latency of 5 and a throughput of 4.
9. Click on Schedule in the Task Bar and expand the Gantt chart.
10. Note that there is a single memory read operation of the “coeffs” singleport memory



11. Parallelism must be added to the design in order to read “data_in” every clock cycle. In this case loop unrolling should be used since the design is already fully pipelined.
12. Click on Architecture in the Task Bar.
13. Click on the MAC loop and set loop unrolling to fully unroll the loop.



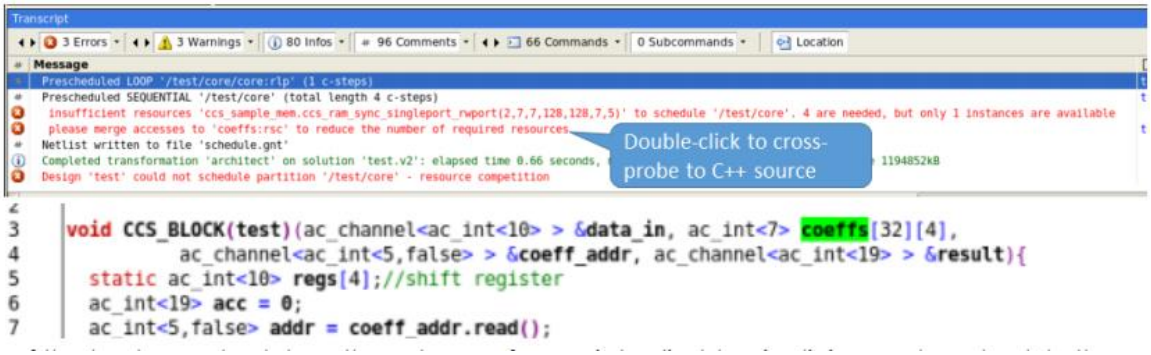
14. Click on RTL in the Task Bar
15. Look at the Catapult Transcript and note:
    a. The tool has generated a resource competition error. This indicates that the pipeline initiation interval (II) has been set so that the design schedule is attempting

to read or write to a memory or IO resource more times in a clock cycle than there are ports on the resource.

b. The error indicates that it needs 4 singleport RAMS, but only 1 is available.

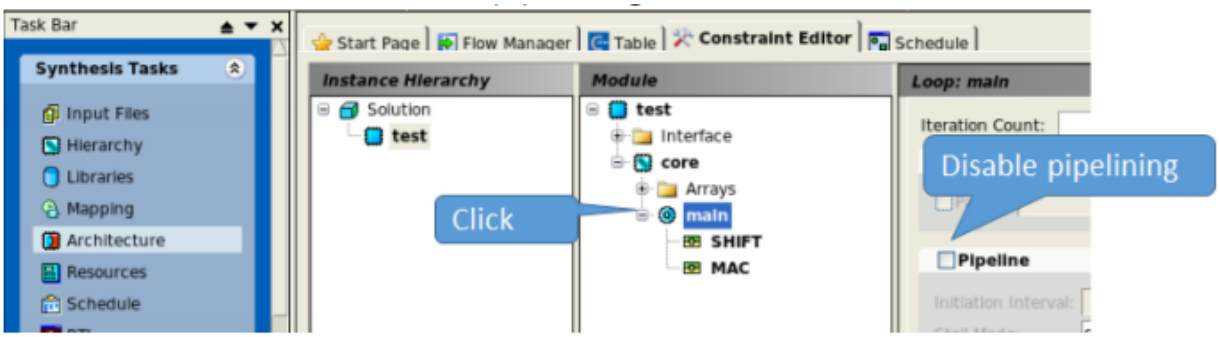
c. The error indicates that the “coeffs_ram” resource is the problem and that the accesses to this memory need to be merged in order to pipeline the design with the current II.

d. Double-click on “please merge accesses to coeffs…” to cross probe to the source and note that the “coeffs” array on the top-level function interface is selected.



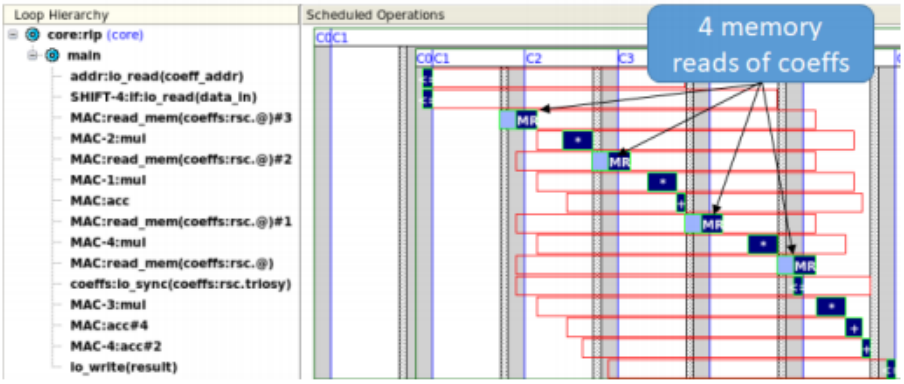
16. One of the best ways to debug these types of errors is to disable pipelining and reschedule the design so that the Gantt chart can be used to analyze the resource bottleneck.

17. Go to architectural constraints and turn off pipelining.



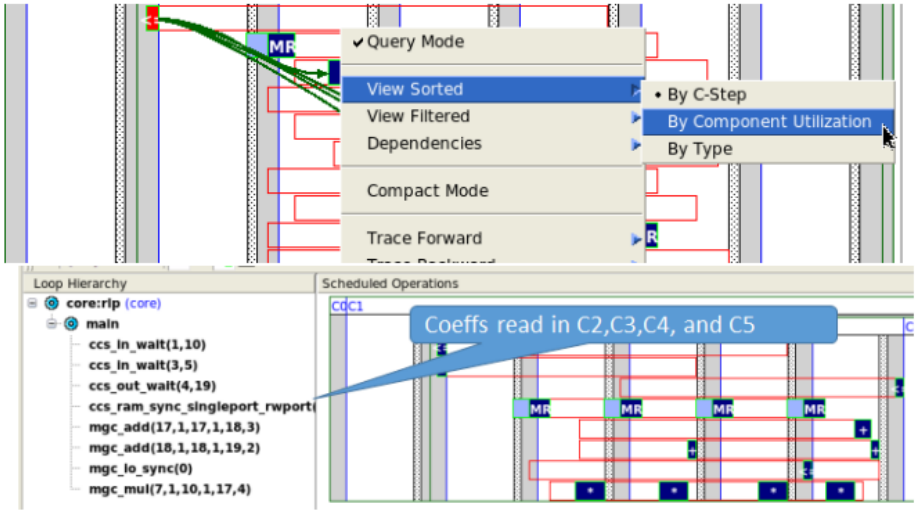
18. Click on Schedule in the Task Bar and expand the Gantt chart.

19. Note that there are 4 memory read operations of “coeffs”.

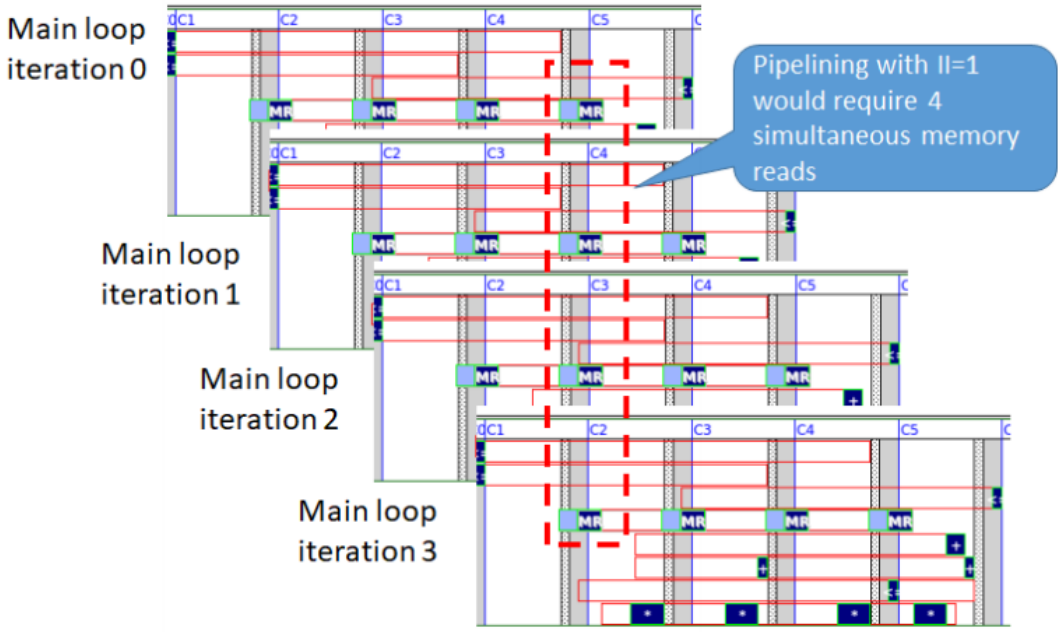


20. Right-click in the Gantt chart and select “View Sorted > By Component Utilization. This will display the schedule so that reads/writes a resource are displayed in the same row.

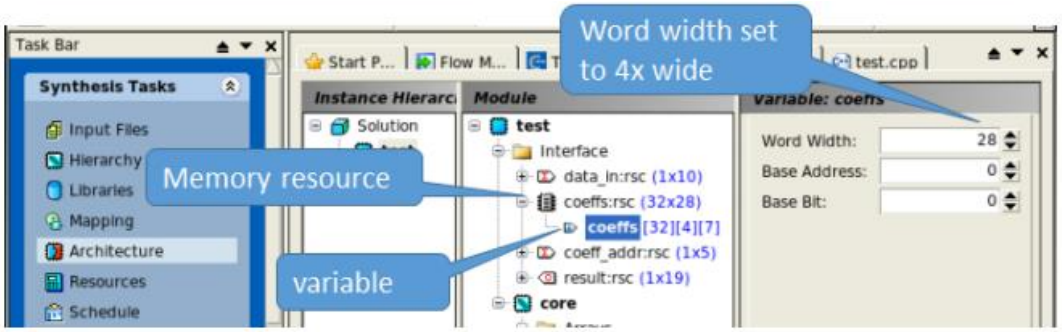
You can see that “coeffs” is read in C2, C3, C4, and C5. Unrolling the MAC look that has the read to coefs[addr][i] has replicated the read 4x.



21. Pipelining the main loop with II=1 means that a new iteration of the main loop must start every clock cycle. This can be visualized by imagining the schedule of the main loop being overlapped so that a new iteration starts every clock cycle. This is illustrated below. It clearly shows that II=1 would require reading the “coeffs” mem 4 times in a c-step or clock cycle.

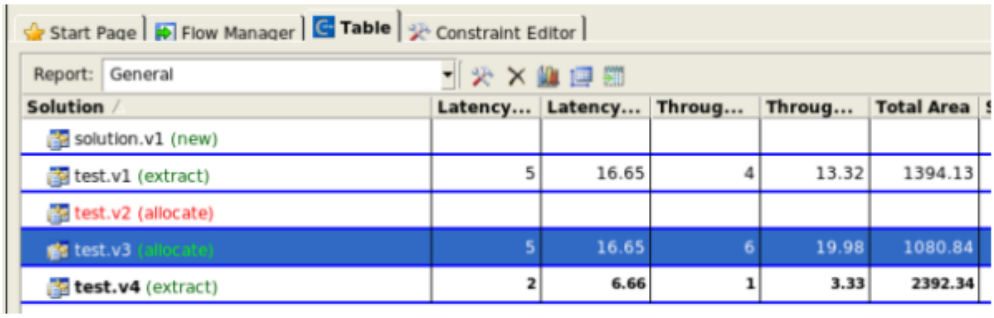


22. Go to architectural constraints and do the following:
    a. Select the coeffs:rsc interface and click on the “+”
    b. Select the “coeffs” variable under the resource and set the word width to 28. This is 4x wider than the data type width of 7-bits. This should allow Catapult to read all 4 values of coeffs used in the MAC loop in parallel from a 4x wide memory.



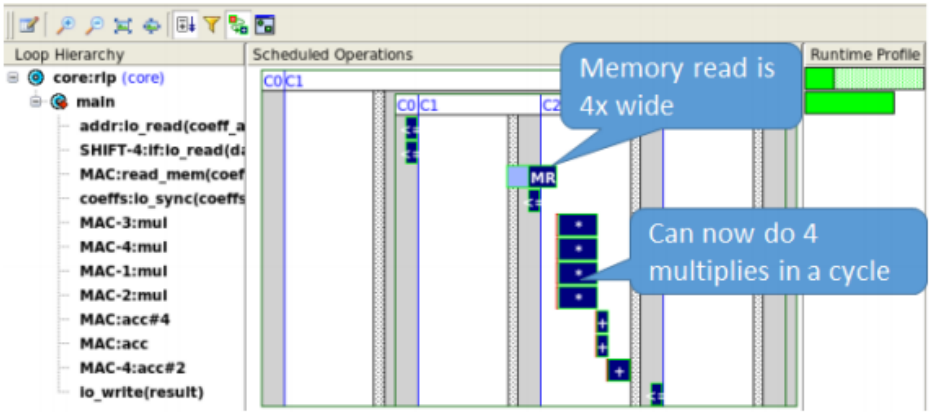
c. Pipeline the Main loop with II=1.

23. Click on RTL in the Task Bar.
24. Look at the Table view and note that the throughput is now equal to 1.

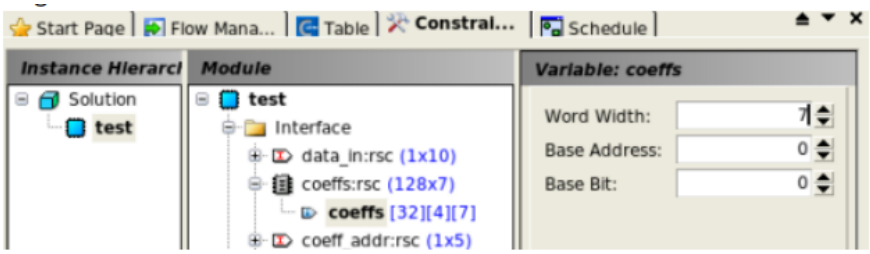


Solution	Latency...	Latency...	Throug...	Throug...	Total Area
solution.v1 (new)					
test.v1 (extract)	5	16.65	4	13.32	1394.13
test.v2 (allocate)					
test.v3 (allocate)	5	16.65	6	19.98	1080.84
test.v4 (extract)	**2**	**6.66**	**1**	**3.33**	**2392.34**

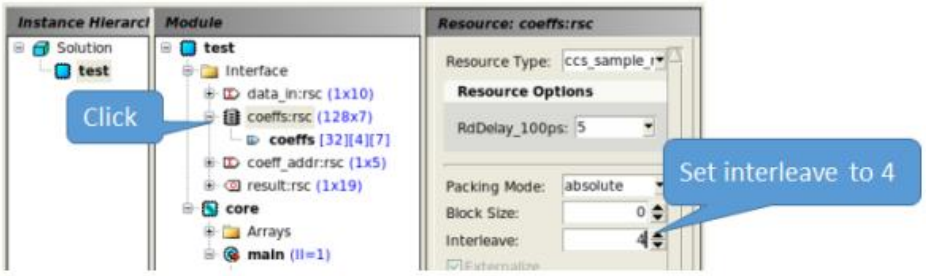
25. Open the Gantt chart and observe that the schedule now has a single memory operation. The 4 memory reads have been merged into a single wide memory read.



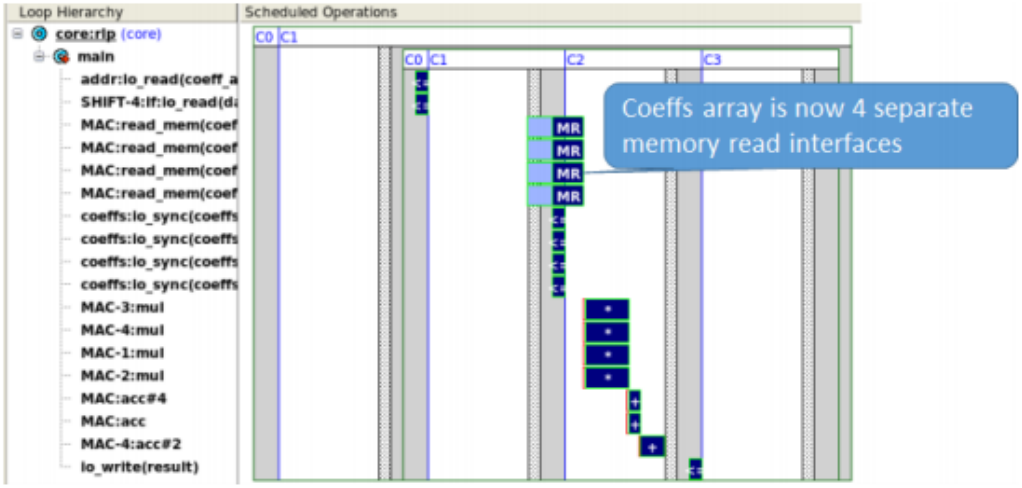
26. Instead of making the memory wider we could have also taken of advantage of the coeffs array accesses. The read of coeffs[addr][i] in the unrolled mac loop means that we are reading coeffs[addr][0], coeffs[addr][1], coeffs[addr][2], and coeffs[addr][3] in parallel. Alternatively we could use interleaving constraints to split the coeffs memory into 4 separate memories.
27. Go to architectural constraints and do the following:
    a. Select the coeffs variable under the coeffs resource and set the word-width back to 7-bits, which is the original width.



b. Click on the coeffs:rsc resource and set interleaving to 4.



28. Click on Schedule in the Task Bar and expand the Gantt chart. The coeffs array has now been “interleaved” into 4 separate memory interfaces.



29. Close Catapult