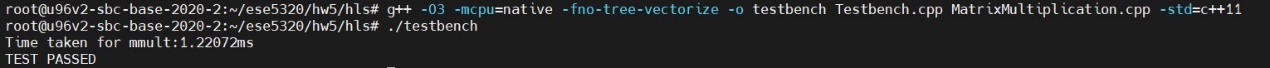
**ESE5320 hw5**

Wirte up by songqij

**Answer**

**Q1**

1.



The time for mmult is 1.22ms

2.

文本

描述已自动生成

3.

Our application involves matrix multiplication. To verify the functionality of the code, we use test input, like how circuits are tested using signals. The Testbench.cpp handles this by providing two test matrices and comparing the actual output with the expected one. When they are the same, they pass the test, while not it fails.

4.

一些文字和图案

描述已自动生成

The expected latency is 19.364~19.419ms.

5.

电脑屏幕的照片上有文字

描述已自动生成

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Name** | BRAM | DSP | FF | LUT | URAM |
| **Total** | 76 | 3 | 7644 | 6217 | 0 |

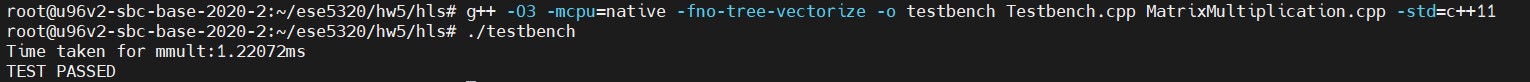
6.

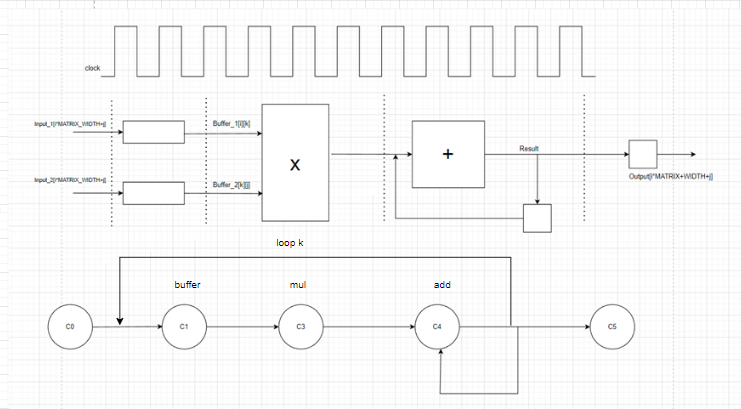
电脑萤幕画面

中度可信度描述已自动生成

The multiplication takes approximately 4 cycles.

7.





8.

The are many factors that slow down the accelerator. Loops are unpipelined and reading and writing to the public memory also takes a relatively long time.

Q2

1.

文本

描述已自动生成

屏幕的截图

描述已自动生成

The latency hasn’t changed from the original one.

电子设备的屏幕

中度可信度描述已自动生成

We can see the difference from the previous one. We can see previous (non-unrolled) one take 10cycles to complete one loop, while unrolled one takes about 20 loops to complete 2 loops. Therefore, the latency is almost the same.

2.

文本

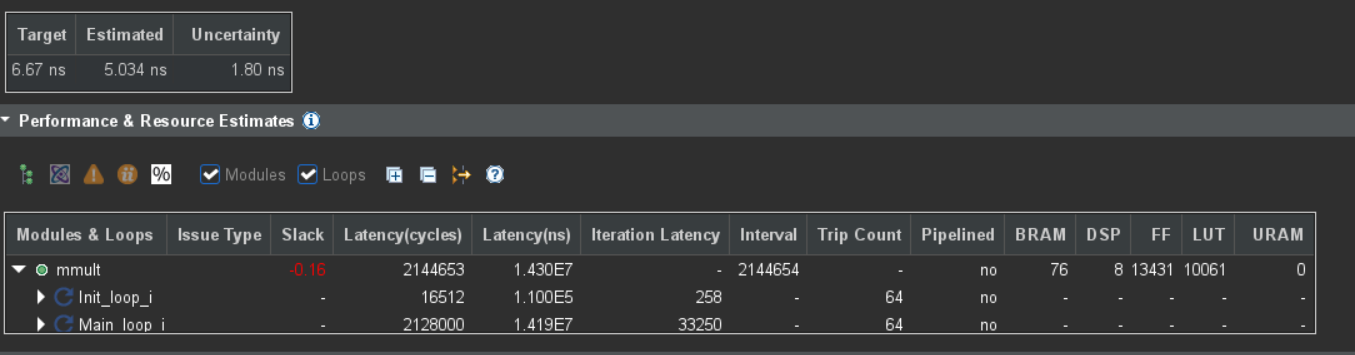
描述已自动生成

3.

We can see from the Schedule Viewer, every time the next Addition will executed after the previous one is finished. Therefore, the fadd has been shared.

Fadd.

4.



It gets a negative slack time, which means that actually we need more times and the clock frequency is too high. A negative slack of -0.16 nanoseconds means that the signal transmission time exceeds the time allocated for the timing path in the design by 0.16 nanoseconds, indicating that the clock cycle is insufficient to accommodate the signal propagation time. It cause the **timing violation.**

5.

表格

描述已自动生成

The expected latency is 13.459~13.541ms

6.

表格

中度可信度描述已自动生成

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Name** | BRAM | DSP | FF | LUT | URAM |
| **Total** | 76 | 8 | 12795 | 9934 | 0 |

7.

**Latency:** Each floating-point addition must complete before the next one begins, which increases the overall latency.

**Non-parallel execution:** Floating-point additions cannot be executed in parallel, which limits the potential throughput of the system.

8.

For 100MHz unroll:

It take 17% of the BRAM\_18K

For 150MHz non-unroll:

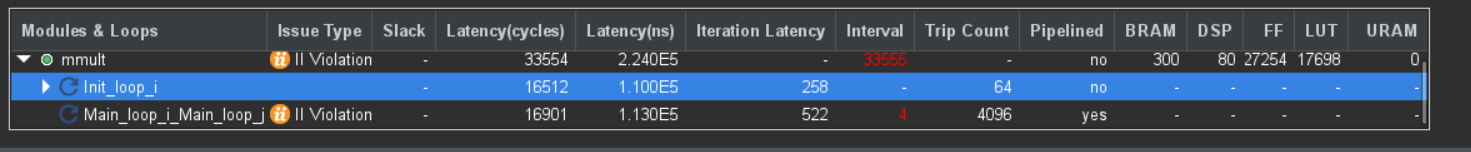
It take 17% of the BRAM\_18K

Therefore these two ways can takes the same copies. However, the 100MHz unroll latency is lower.

I will choose unroll 100MHz.

Q3

1.



The interval is 4

2.

图示

描述已自动生成

3.

It should be 64 per second.

4.

First of all, the loading speed is too low, every time it could only load 16bytes and not enough for the execution.

And for adding, it adds sequentially, so that we could not achieve the II of 1.

5.

I divided the buffer\_1 which is 64\*64 into 64\* 1x64

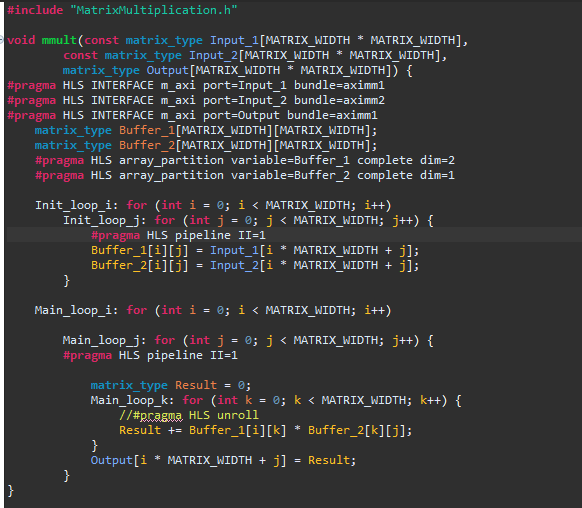
I divided the buffer\_2 which is 64\*64 into 64\* 64x1

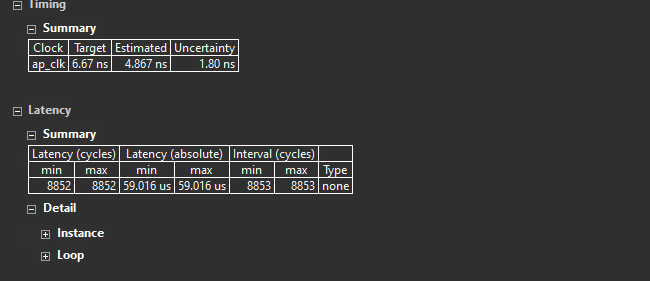
And in one loop iteration will multiply 1x64 and 64x1 these two buffers.

图示

描述已自动生成

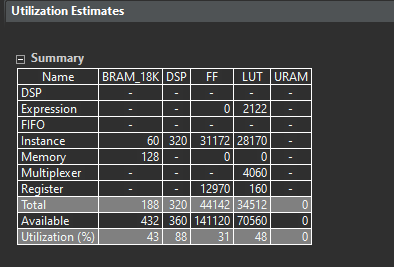
6.





The expected latency is 0.059016ms

7.

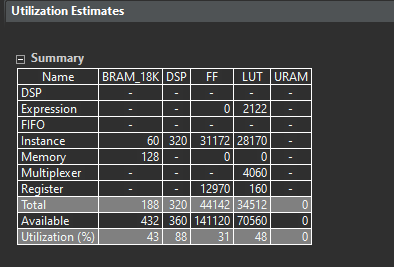
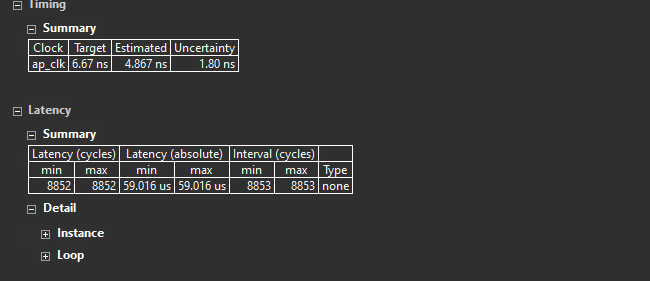
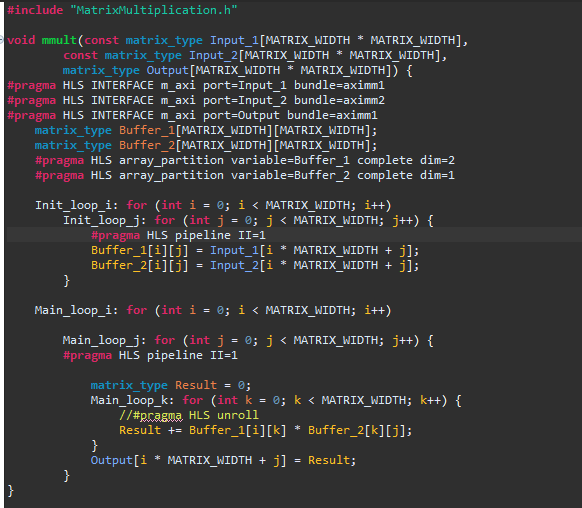


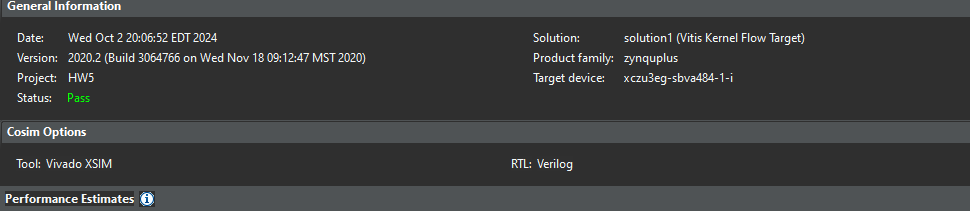
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Name** | BRAM | DSP | FF | LUT | URAM |
| **Total** | 188 | 320 | 44142 | 34512 | 0 |

Yes, it is pretty consuming on DSP usage as it gets the 88% of Utilization.

And for BRAM, flip flop and LUT all reached nearly 50% of Utilization.

8.



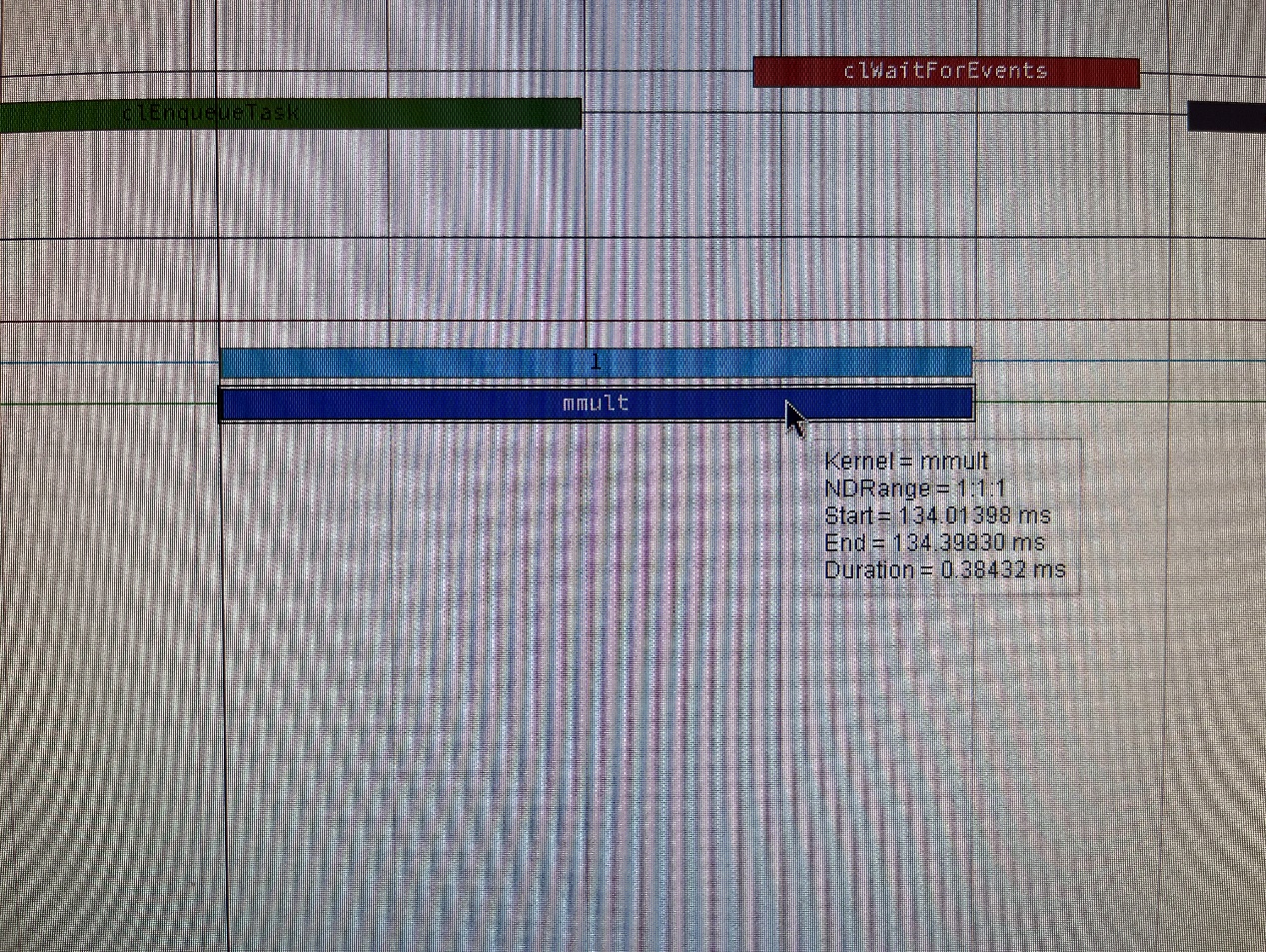


9.Done

Q4

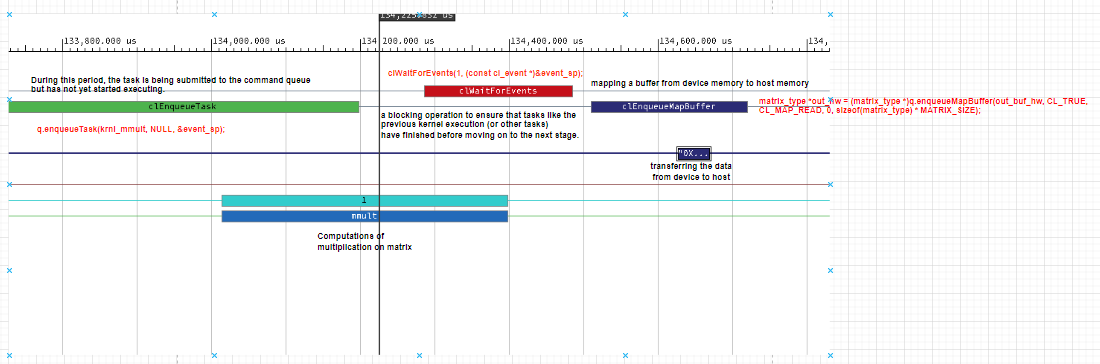
1.Done

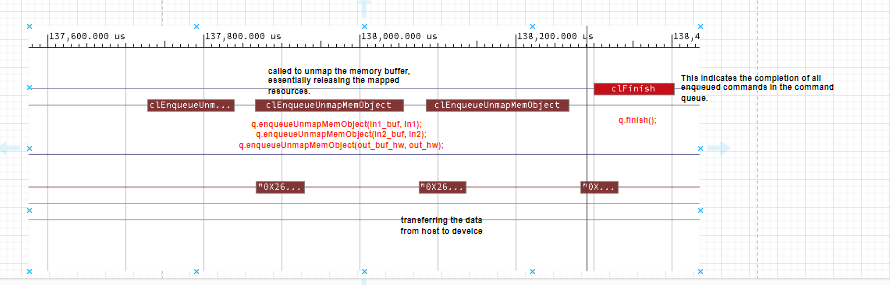
2.



The latency of mmult is 0.38432ms

3.



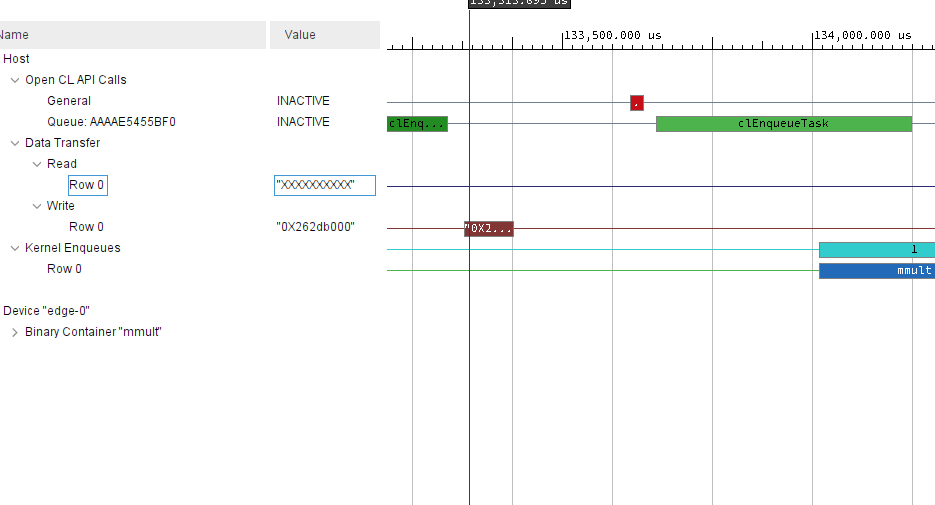


Q5.

1.

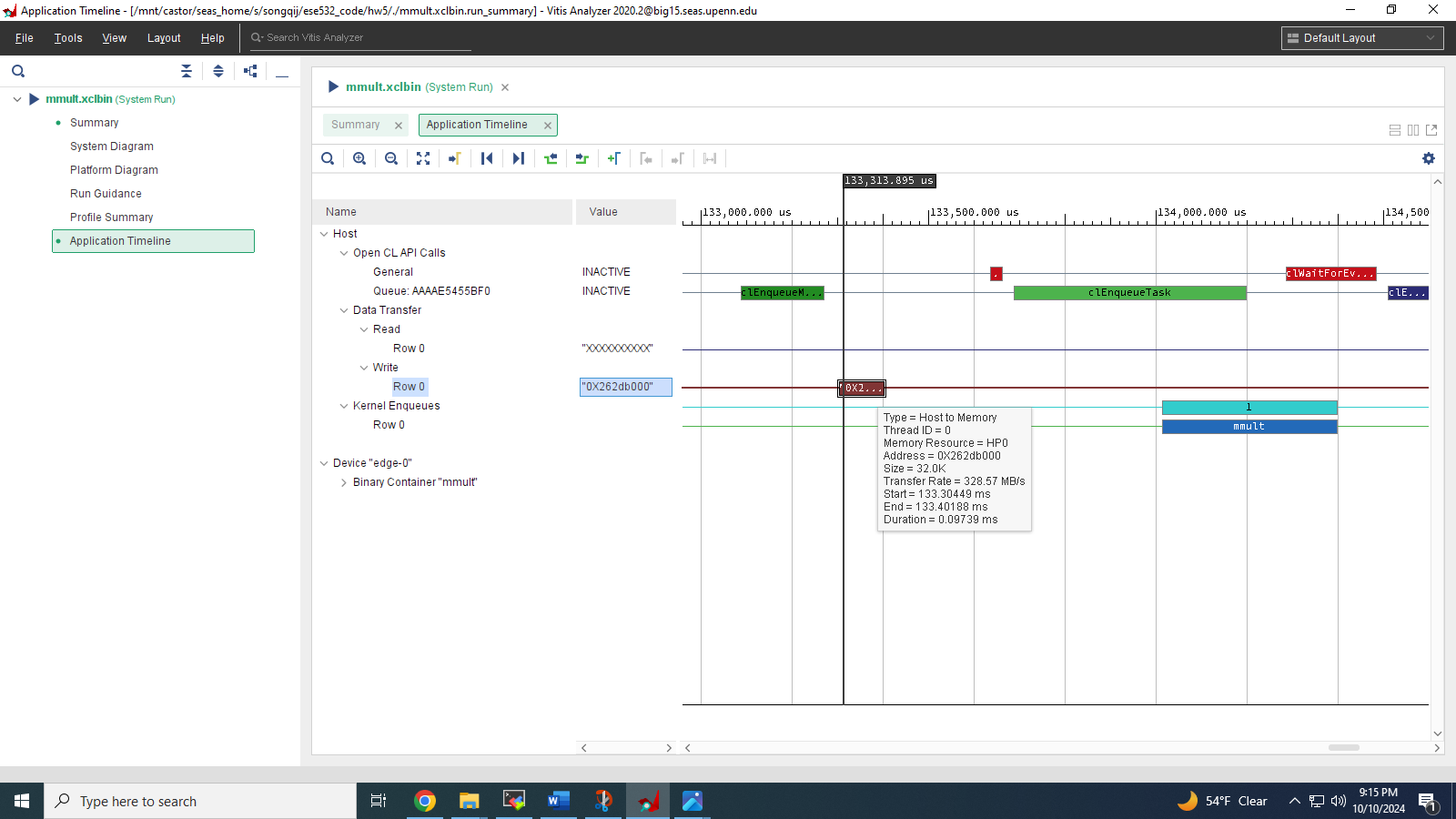
2.

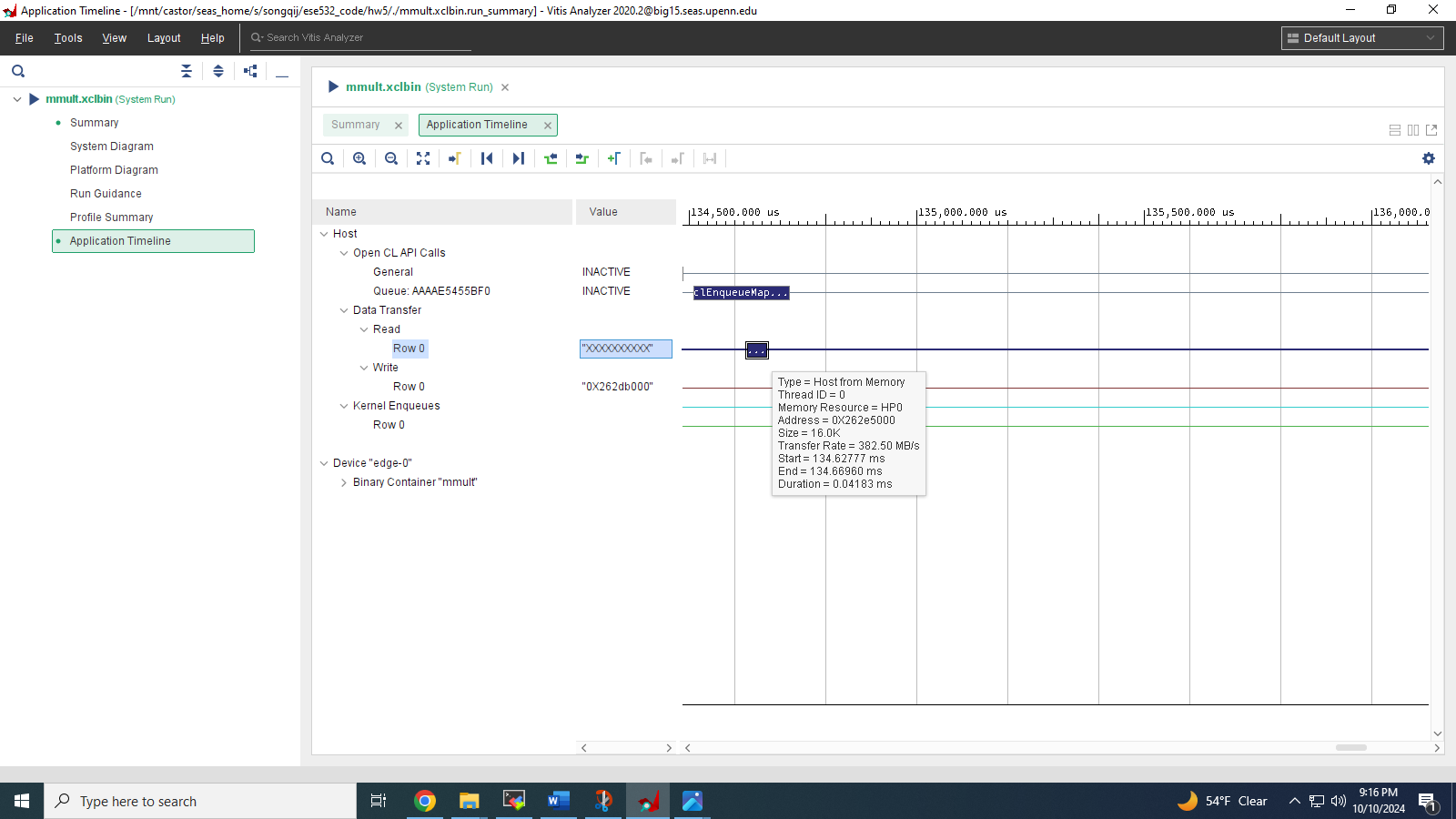
For we are calculating T setup, we just add up all the time before mmult including space time.

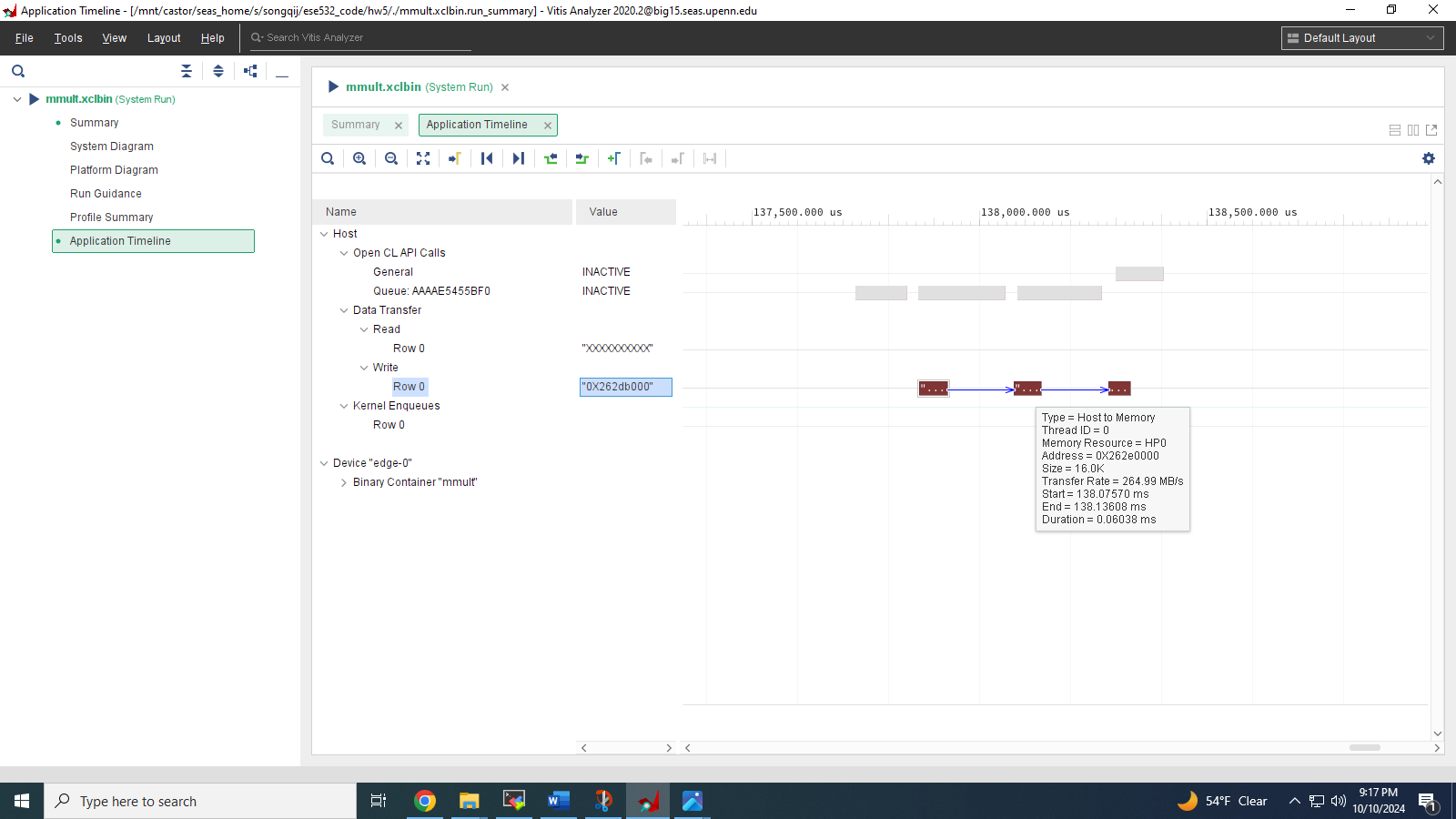


3.

For calculating the , just add up all the host from memory and host to memory.







Therefore,

4.

should be larger than 134.197ms.

5.

For

6.

For the main loop j pipelined and loop k unrolled the should be:

7.

The transferring time should be the reading adds to the writing part.

The reading time with two buffers are

The writing time with the buffer is

8.

9.

10.

11.

Q6.

For question one we measure the baseline time for the matrix multiplication.

For question two we do the unroll of the main loop k and do the optimization.

For question three we do the pipeline of the main loop j and the initial loop j, and get the final optimization.

2.

