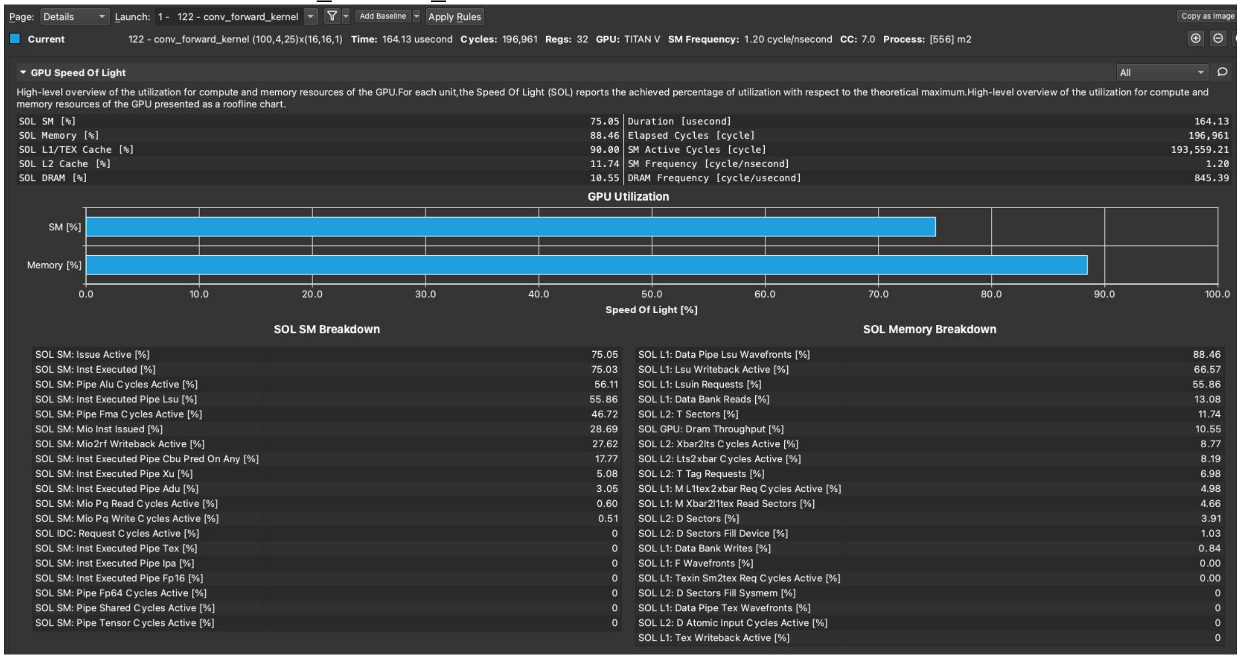
Name: Yiming Li
NetID: Yiming22
Section: AL1

ECE 408/CS483 Milestone 3 Report

0. List Op Times, whole program execution time, and accuracy for batch size of 100, 1k, and 10k images from your basic forward convolution kernel in milestone 2. This will act as your baseline this milestone.

Batch Size	Op Time 1	Op Time 2	Total Execution Time	Accuracy
100	0.175098 ms	0.632573 ms	0m1.190s	0.86
1000	1.63944 ms	6.2645 ms	0m9.660s	0.886
10000	16.0852 ms	62.7071 ms	1m37.766s	0.8714

Launch: 1 - 122 - conv_forward_kernel



Current 122 - conv_forward_kernel (100,4,25)x(16,16,1) **Time:** 164.13 usecond **Cycles:** 196,961 **Regs:** 32 **GPU:** TITAN V **SM Frequency:** 1.20 cycle/nsecond **CC:** 7.0 **Process:** [556] m2

GPU Speed Of Light

High-level overview of the utilization for compute and memory resources of the GPU.For each unit,the Speed Of Light (SOL) reports the achieved percentage of utilization with respect to the theoretical maximum.High-level overview of the utilization for compute and memory resources of the GPU presented as a roofline chart.

```
SOL SM [%]            75.05 | Duration [usecond]              164.13
SOL Memory [%]        88.46 | Elapsed Cycles [cycle]         196,961
SOL L1/TEX Cache [%]  90.00 | SM Active Cycles [cycle]    193,559.21
SOL L2 Cache [%]      11.74 | SM Frequency [cycle/nsecond]      1.20
SOL DRAM [%]          10.55 | DRAM Frequency [cycle/usecond]  845.39
```

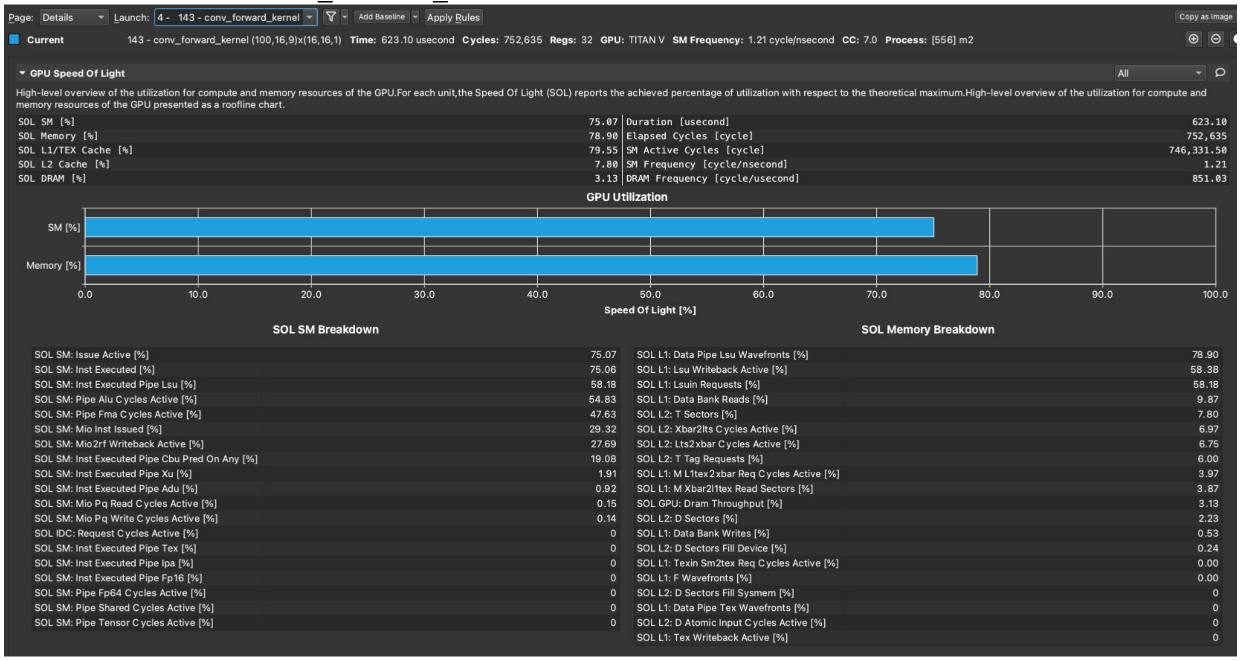
SOL SM Breakdown

SOL SM: Issue Active [%]	75.05
SOL SM: Inst Executed [%]	75.03
SOL SM: Pipe Alu Cycles Active [%]	56.11
SOL SM: Inst Executed Pipe Lsu [%]	55.86
SOL SM: Pipe Fma Cycles Active [%]	46.72
SOL SM: Mio Inst Issued [%]	28.69
SOL SM: Mio2rf Writeback Active [%]	27.62
SOL SM: Inst Executed Pipe Cbu Pred On Any [%]	17.77
SOL SM: Inst Executed Pipe Xu [%]	5.08
SOL SM: Inst Executed Pipe Adu [%]	3.05
SOL SM: Mio Pq Read Cycles Active [%]	0.60
SOL SM: Mio Pq Write Cycles Active [%]	0.51
SOL IDC: Request Cycles Active [%]	0
SOL SM: Inst Executed Pipe Tex [%]	0
SOL SM: Inst Executed Pipe Ipa [%]	0
SOL SM: Inst Executed Pipe Fp16 [%]	0
SOL SM: Pipe Fp64 Cycles Active [%]	0
SOL SM: Pipe Shared Cycles Active [%]	0
SOL SM: Pipe Tensor Cycles Active [%]	0

SOL Memory Breakdown

SOL L1: Data Pipe Lsu Wavefronts [%]	88.46
SOL L1: Lsu Writeback Active [%]	66.57
SOL L1: Lsuin Requests [%]	55.86
SOL L1: Data Bank Reads [%]	13.08
SOL L2: T Sectors [%]	11.74
SOL GPU: Dram Throughput [%]	10.55
SOL L2: Xbar2lts Cycles Active [%]	8.77
SOL L2: Lts2xbar Cycles Active [%]	8.19
SOL L2: T Tag Requests [%]	6.98
SOL L1: M L1tex2xbar Req Cycles Active [%]	4.98
SOL L1: M Xbar2l1tex Read Sectors [%]	4.66
SOL L2: D Sectors [%]	3.91
SOL L2: D Sectors Fill Device [%]	1.03
SOL L1: Data Bank Writes [%]	0.84
SOL L1: F Wavefronts [%]	0.00
SOL L1: Texin Sm2tex Req Cycles Active [%]	0.00
SOL L2: D Sectors Fill Sysmem [%]	0
SOL L1: Data Pipe Tex Wavefronts [%]	0
SOL L2: D Atomic Input Cycles Active [%]	0
SOL L1: Tex Writeback Active [%]	0

Launch: 4 -   143 - conv_forward_kernel



Current 143 - conv_forward_kernel (100,16,9)x(16,16,1) **Time:** 623.10 usecond **Cycles:** 752,635 **Regs:** 32 **GPU:** TITAN V **SM Frequency:** 1.21 cycle/nsecond **CC:** 7.0 **Process:** [556] m2

High-level overview of the utilization for compute and memory resources of the GPU.For each unit,the Speed Of Light (SOL) reports the achieved percentage of utilization with respect to the theoretical maximum.High-level overview of the utilization for compute and memory resources of the GPU presented as a roofline chart.

SOL SM [%]	75.07	Duration [usecond]	623.10
SOL Memory [%]	78.90	Elapsed Cycles [cycle]	752,635
SOL L1/TEX Cache [%]	79.55	SM Active Cycles [cycle]	746,331.50
SOL L2 Cache [%]	7.80	SM Frequency [cycle/nsecond]	1.21
SOL DRAM [%]	3.13	DRAM Frequency [cycle/usecond]	851.03

SOL SM Breakdown

SOL SM: Issue Active [%]	75.07
SOL SM: Inst Executed [%]	75.06
SOL SM: Inst Executed Pipe Lsu [%]	58.18
SOL SM: Pipe Alu Cycles Active [%]	54.83
SOL SM: Pipe Fma Cycles Active [%]	47.63
SOL SM: Mio Inst Issued [%]	29.32
SOL SM: Mio2rf Writeback Active [%]	27.69
SOL SM: Inst Executed Pipe Cbu Pred On Any [%]	19.08
SOL SM: Inst Executed Pipe Xu [%]	1.91
SOL SM: Inst Executed Pipe Adu [%]	0.92
SOL SM: Mio Pq Read Cycles Active [%]	0.15
SOL SM: Mio Pq Write Cycles Active [%]	0.14
SOL IDC: Request Cycles Active [%]	0
SOL SM: Inst Executed Pipe Tex [%]	0
SOL SM: Inst Executed Pipe Ipa [%]	0
SOL SM: Inst Executed Pipe Fp16 [%]	0
SOL SM: Pipe Fp64 Cycles Active [%]	0
SOL SM: Pipe Shared Cycles Active [%]	0
SOL SM: Pipe Tensor Cycles Active [%]	0

SOL Memory Breakdown

SOL L1: Data Pipe Lsu Wavefronts [%]	78.90
SOL L1: Lsu Writeback Active [%]	58.38
SOL L1: Lsuin Requests [%]	58.18
SOL L1: Data Bank Reads [%]	9.87
SOL L2: T Sectors [%]	7.80
SOL L2: Xbar2lts Cycles Active [%]	6.97
SOL L2: Lts2xbar Cycles Active [%]	6.75
SOL L2: T Tag Requests [%]	6.00
SOL L1: M L1tex2xbar Req Cycles Active [%]	3.97
SOL L1: M Xbar2l1tex Read Sectors [%]	3.87
SOL GPU: Dram Throughput [%]	3.13
SOL L2: D Sectors [%]	2.23
SOL L1: Data Bank Writes [%]	0.53
SOL L2: D Sectors Fill Device [%]	0.24
SOL L1: Texin Sm2tex Req Cycles Active [%]	0.00
SOL L1: F Wavefronts [%]	0.00
SOL L2: D Sectors Fill Sysmem [%]	0
SOL L1: Data Pipe Tex Wavefronts [%]	0
SOL L2: D Atomic Input Cycles Active [%]	0
SOL L1: Tex Writeback Active [%]	0

Baseline:
The first conv_forward_kernel GPU utilization SM is 75.05%, the Memory is 88.46%.
The second conv_forward_kernel GPU utilization SM is 75.07%, the Memory is 78.90%.

1. Optimization 1: Weight matrix (kernel values) in constant memory

    a. Which optimization did you choose to implement and why did you choose that optimization technique.

    Weight matrix (kernel values) in constant memory (1 point)

    b. How does the optimization work? Did you think the optimization would increase performance of the forward convolution? Why?
    Does the optimization synergize with any of your previous optimizations?

    The weighted matrix is never changed in this project. And if we move the matrix to constant memory, I think it can speed up the operation time and can get better use of bandwidth.

    No synergy.

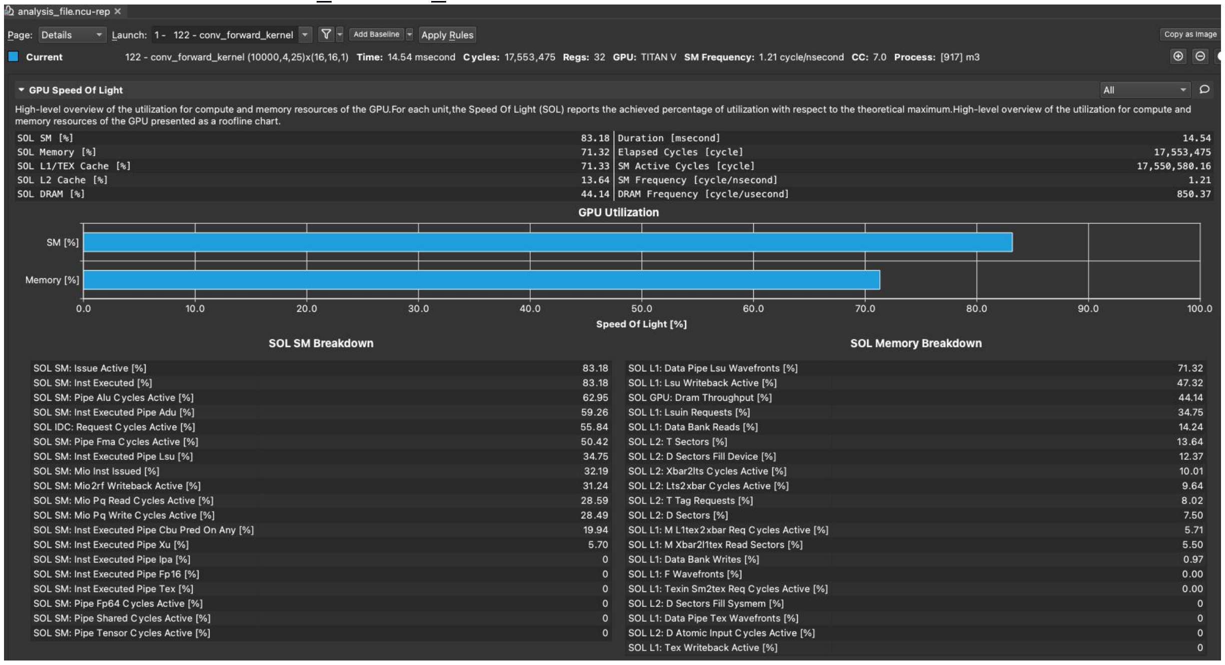
c. List the Op Times, whole program execution time, and accuracy for batch size of 100, 1k, and 10k images using this optimization (including any previous optimizations also used).

Batch Size	Op Time 1	Op Time 2	Total Execution Time	Accuracy
100	0.165795 ms	0.57739 ms	0m1.595s	0.86
1000	1.48544 ms	5.65347 ms	0m9.798s	0.886
10000	14.5932 ms	14.5932 ms	1m38.642s	0.8714

d. Was implementing this optimization successful in improving performance? Why or why not? Include profiling results from nsys and Nsight-Compute to justify your answer, directly comparing to your baseline (or the previous optimization this one is built off of).

I think it's successful to optimization.
profiling results from nsys and Nsight-Compute:

Launch: 1 -   122 - conv_forward_kernel



Current 122 - conv_forward_kernel (10000,4,25)x(16,16,1) **Time:** 14.54 msecond **Cycles:** 17,553,475 **Regs:** 32 **GPU:** TITAN V **SM Frequency:** 1.21 cycle/nsecond **CC:** 7.0 **Process:** [917] m3

GPU Speed Of Light

High-level overview of the utilization for compute and memory resources of the GPU.For each unit,the Speed Of Light (SOL) reports the achieved percentage of utilization with respect to the theoretical maximum.High-level overview of the utilization for compute and memory resources of the GPU presented as a roofline chart.

SOL SM [%]	83.18	Duration [msecond]	14.54
SOL Memory [%]	71.32	Elapsed Cycles [cycle]	17,553,475
SOL L1/TEX Cache [%]	71.33	SM Active Cycles [cycle]	17,550,580.16
SOL L2 Cache [%]	13.64	SM Frequency [cycle/nsecond]	1.21
SOL DRAM [%]	44.14	DRAM Frequency [cycle/usecond]	850.37

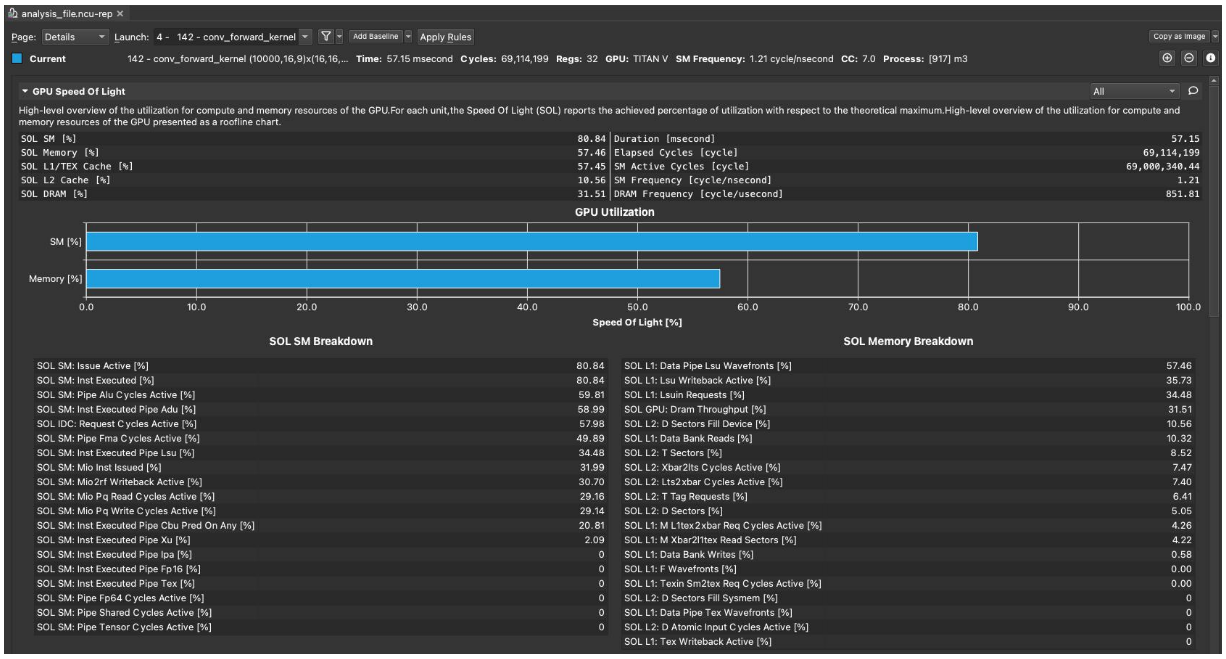
SOL SM Breakdown

SOL SM: Issue Active [%]	83.18
SOL SM: Inst Executed [%]	83.18
SOL SM: Pipe Alu Cycles Active [%]	62.95
SOL SM: Inst Executed Pipe Adu [%]	59.26
SOL IDC: Request Cycles Active [%]	55.84
SOL SM: Pipe Fma Cycles Active [%]	50.42
SOL SM: Inst Executed Pipe Lsu [%]	34.75
SOL SM: Mio Inst Issued [%]	32.19
SOL SM: Mio2rf Writeback Active [%]	31.24
SOL SM: Mio Pq Read Cycles Active [%]	28.59
SOL SM: Mio Pq Write Cycles Active [%]	28.49
SOL SM: Inst Executed Pipe Cbu Pred On Any [%]	19.94
SOL SM: Inst Executed Pipe Xu [%]	5.70
SOL SM: Inst Executed Pipe Ipa [%]	0
SOL SM: Inst Executed Pipe Fp16 [%]	0
SOL SM: Inst Executed Pipe Tex [%]	0
SOL SM: Pipe Fp64 Cycles Active [%]	0
SOL SM: Pipe Shared Cycles Active [%]	0
SOL SM: Pipe Tensor Cycles Active [%]	0

SOL Memory Breakdown

SOL L1: Data Pipe Lsu Wavefronts [%]	71.32
SOL L1: Lsu Writeback Active [%]	47.32
SOL GPU: Dram Throughput [%]	44.14
SOL L1: Lsuin Requests [%]	34.75
SOL L1: Data Bank Reads [%]	14.24
SOL L2: T Sectors [%]	13.64
SOL L2: D Sectors Fill Device [%]	12.37
SOL L2: Xbar2lts Cycles Active [%]	10.01
SOL L2: Lts2xbar Cycles Active [%]	9.64
SOL L2: T Tag Requests [%]	8.02
SOL L2: D Sectors [%]	7.50
SOL L1: M L1tex2xbar Req Cycles Active [%]	5.71
SOL L1: M Xbar2l1tex Read Sectors [%]	5.50
SOL L1: Data Bank Writes [%]	0.97
SOL L1: F Wavefronts [%]	0.00
SOL L1: Texin Sm2tex Req Cycles Active [%]	0.00
SOL L2: D Sectors Fill Sysmem [%]	0
SOL L1: Data Pipe Tex Wavefronts [%]	0
SOL L2: D Atomic Input Cycles Active [%]	0
SOL L1: Tex Writeback Active [%]	0

Launch: 4 -   142 - conv_forward_kernel


analysis_file.ncu-rep

Page: Details Launch: 4 - 142 - conv_forward_kernel Add Baseline Apply Rules Copy as Image

Current 142 - conv_forward_kernel (10000,16,9)x(16,16,... **Time:** 57.15 msecond **Cycles:** 69,114,199 **Regs:** 32 **GPU:** TITAN V **SM Frequency:** 1.21 cycle/nsecond **CC:** 7.0 **Process:** [917] m3

GPU Speed Of Light

High-level overview of the utilization for compute and memory resources of the GPU.For each unit,the Speed Of Light (SOL) reports the achieved percentage of utilization with respect to the theoretical maximum.High-level overview of the utilization for compute and memory resources of the GPU presented as a roofline chart.

SOL SM [%]	80.84	Duration [msecond]	57.15
SOL Memory [%]	57.46	Elapsed Cycles [cycle]	69,114,199
SOL L1/TEX Cache [%]	57.45	SM Active Cycles [cycle]	69,000,340.44
SOL L2 Cache [%]	10.56	SM Frequency [cycle/nsecond]	1.21
SOL DRAM [%]	31.51	DRAM Frequency [cycle/usecond]	851.81

SOL SM Breakdown

SOL SM: Issue Active [%]	80.84
SOL SM: Inst Executed [%]	80.84
SOL SM: Pipe Alu Cycles Active [%]	59.81
SOL SM: Inst Executed Pipe Adu [%]	58.99
SOL IDC: Request Cycles Active [%]	57.98
SOL SM: Pipe Fma Cycles Active [%]	49.89
SOL SM: Inst Executed Pipe Lsu [%]	34.48
SOL SM: Mio Inst Issued [%]	31.99
SOL SM: Mio2rf Writeback Active [%]	30.70
SOL SM: Mio Pq Read Cycles Active [%]	29.16
SOL SM: Mio Pq Write Cycles Active [%]	29.14
SOL SM: Inst Executed Pipe Cbu Pred On Any [%]	20.81
SOL SM: Inst Executed Pipe Xu [%]	2.09
SOL SM: Inst Executed Pipe Ipa [%]	0
SOL SM: Inst Executed Pipe Fp16 [%]	0
SOL SM: Inst Executed Pipe Tex [%]	0
SOL SM: Pipe Fp64 Cycles Active [%]	0
SOL SM: Pipe Shared Cycles Active [%]	0
SOL SM: Pipe Tensor Cycles Active [%]	0

SOL Memory Breakdown

SOL L1: Data Pipe Lsu Wavefronts [%]	57.46
SOL L1: Lsu Writeback Active [%]	35.73
SOL L1: Lsuin Requests [%]	34.48
SOL GPU: Dram Throughput [%]	31.51
SOL L2: D Sectors Fill Device [%]	10.56
SOL L1: Data Bank Reads [%]	10.32
SOL L2: T Sectors [%]	8.52
SOL L2: Xbar2lts Cycles Active [%]	7.47
SOL L2: Lts2xbar Cycles Active [%]	7.40
SOL L2: T Tag Requests [%]	6.41
SOL L2: D Sectors [%]	5.05
SOL L1: M L1tex2xbar Req Cycles Active [%]	4.26
SOL L1: M Xbar2l1tex Read Sectors [%]	4.22
SOL L1: Data Bank Writes [%]	0.58
SOL L1: F Wavefronts [%]	0.00
SOL L1: Texin Sm2tex Req Cycles Active [%]	0.00
SOL L2: D Sectors Fill Sysmem [%]	0
SOL L1: Data Pipe Tex Wavefronts [%]	0
SOL L2: D Atomic Input Cycles Active [%]	0
SOL L1: Tex Writeback Active [%]	0

The first conv_forward_kernel GPU utilization SM is 83.18%(compared to base line: 75.05%), the Memory is 71.32%(compared with base line: 88.46%)
The second conv_forward_kernel GPU utilization SM is 80.84% (compared to base line: 75.07%), the Memory is 57.46%(compared with base line: 78.90%)
The constant memory help kernel to access less global memory and the time of this is cut down largely.

e. What references did you use when implementing this technique?

   Constant memory in C language, chapter 7

2. Optimization 2: Tuning with restrict and loop unrolling

   a. Which optimization did you choose to implement and why did you choose that optimization technique.

      Tuning with restrict and loop unrolling (considered as one optimization only if you do both) (3 points)

b. How does the optimization work? Did you think the optimization would increase performance of the forward convolution? Why? Does the optimization synergize with any of your previous optimizations?

   Loop unrolling can help optimizing the execution time. It speeds up the program by getting rid of the control/test instructions in the loop. This is implemented by removing and reducing number of iterations.
   No synergy.

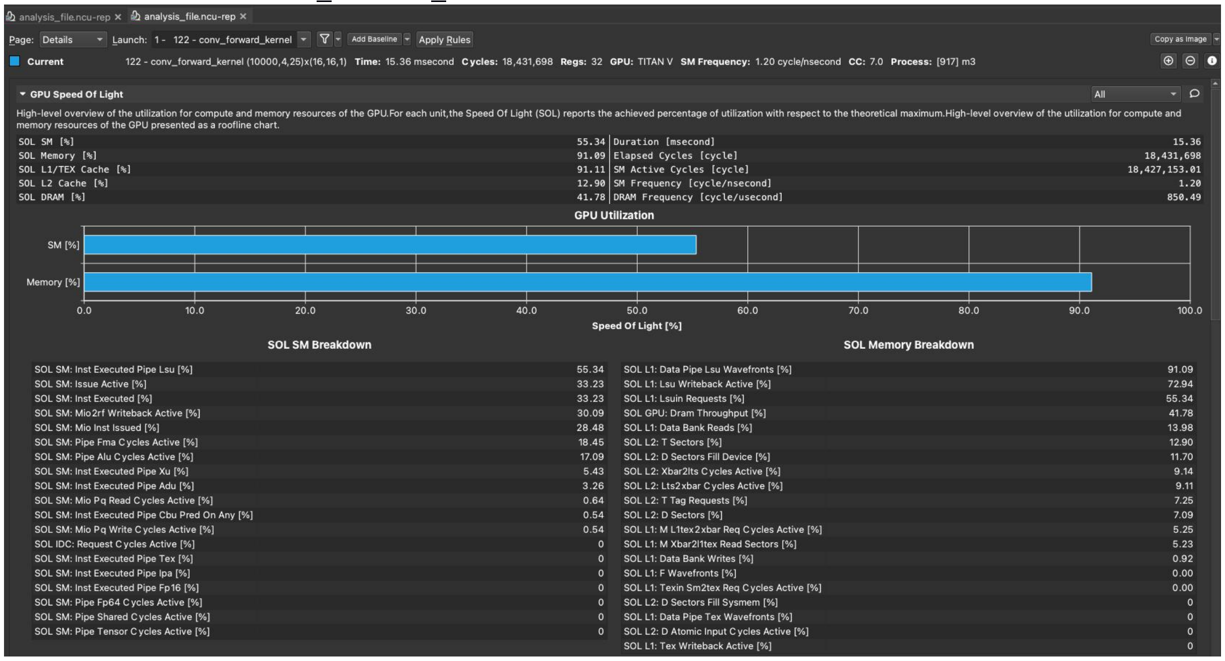
c. List the Op Times, whole program execution time, and accuracy for batch size of 100, 1k, and 10k images using this optimization (including any previous optimizations also used).

Batch Size	Op Time 1	Op Time 2	Total Execution Time	Accuracy
100	0.167156 ms	0.598132 ms	0m1.387s	0.86
1000	1.5526 ms	5.97613 ms	0m10.103s	0.886
10000	15.382 ms	60.2823 ms	1m43.120s	0.8714

d. Was implementing this optimization successful in improving performance? Why or why not? Include profiling results from nsys and Nsight-Compute to justify your answer, directly comparing to your baseline (or the previous optimization this one is built off of).

I think it’s successful to optimization.
profiling results from nsys and Nsight-Compute:

Launch: 1 -   122 - conv_forward_kernel



Current   122 - conv_forward_kernel (10000,4,25)x(16,16,1) **Time:** 15.36 msecond **Cycles:** 18,431,698 **Regs:** 32 **GPU:** TITAN V **SM Frequency:** 1.20 cycle/nsecond **CC:** 7.0 **Process:** [917] m3

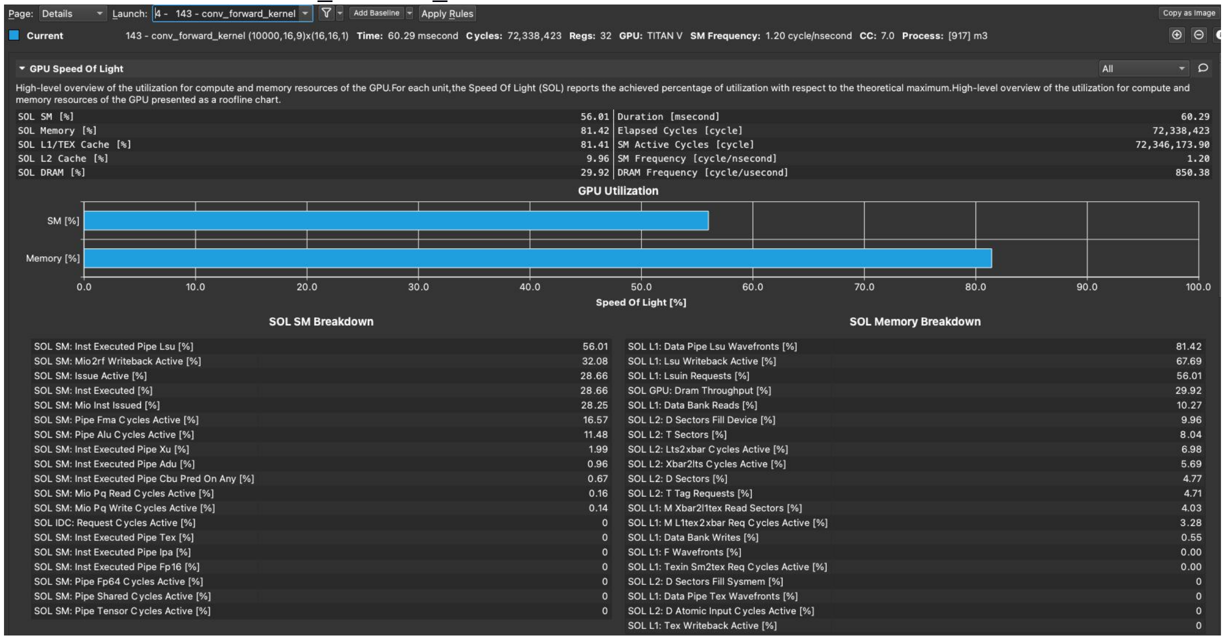
GPU Speed Of Light

High-level overview of the utilization for compute and memory resources of the GPU.For each unit,the Speed Of Light (SOL) reports the achieved percentage of utilization with respect to the theoretical maximum.High-level overview of the utilization for compute and memory resources of the GPU presented as a roofline chart.

```
SOL SM [%]                 55.34 | Duration [msecond]                    15.36
SOL Memory [%]             91.09 | Elapsed Cycles [cycle]           18,431,698
SOL L1/TEX Cache [%]       91.11 | SM Active Cycles [cycle]      18,427,153.01
SOL L2 Cache [%]           12.90 | SM Frequency [cycle/nsecond]           1.20
SOL DRAM [%]               41.78 | DRAM Frequency [cycle/usecond]       850.49
```

SOL SM Breakdown		SOL Memory Breakdown	
SOL SM: Inst Executed Pipe Lsu [%]	55.34	SOL L1: Data Pipe Lsu Wavefronts [%]	91.09
SOL SM: Issue Active [%]	33.23	SOL L1: Lsu Writeback Active [%]	72.94
SOL SM: Inst Executed [%]	33.23	SOL L1: Lsuin Requests [%]	55.34
SOL SM: Mio2rf Writeback Active [%]	30.09	SOL GPU: Dram Throughput [%]	41.78
SOL SM: Mio Inst Issued [%]	28.48	SOL L1: Data Bank Reads [%]	13.98
SOL SM: Pipe Fma Cycles Active [%]	18.45	SOL L2: T Sectors [%]	12.90
SOL SM: Pipe Alu Cycles Active [%]	17.09	SOL L2: D Sectors Fill Device [%]	11.70
SOL SM: Inst Executed Pipe Xu [%]	5.43	SOL L2: Xbar2lts Cycles Active [%]	9.14
SOL SM: Inst Executed Pipe Adu [%]	3.26	SOL L2: Lts2xbar Cycles Active [%]	9.11
SOL SM: Mio Pq Read Cycles Active [%]	0.64	SOL L2: T Tag Requests [%]	7.25
SOL SM: Inst Executed Pipe Cbu Pred On Any [%]	0.54	SOL L2: D Sectors [%]	7.09
SOL SM: Mio Pq Write Cycles Active [%]	0.54	SOL L1: M L1tex2xbar Req Cycles Active [%]	5.25
SOL IDC: Request Cycles Active [%]	0	SOL L1: M Xbar2l1tex Read Sectors [%]	5.23
SOL SM: Inst Executed Pipe Tex [%]	0	SOL L1: Data Bank Writes [%]	0.92
SOL SM: Inst Executed Pipe Ipa [%]	0	SOL L1: F Wavefronts [%]	0.00
SOL SM: Inst Executed Pipe Fp16 [%]	0	SOL L1: Texin Sm2tex Req Cycles Active [%]	0.00
SOL SM: Pipe Fp64 Cycles Active [%]	0	SOL L2: D Sectors Fill Sysmem [%]	0
SOL SM: Pipe Shared Cycles Active [%]	0	SOL L1: Data Pipe Tex Wavefronts [%]	0
SOL SM: Pipe Tensor Cycles Active [%]	0	SOL L2: D Atomic Input Cycles Active [%]	0
		SOL L1: Tex Writeback Active [%]	0

Launch: 4 -   143 - conv_forward_kernel



Current   143 - conv_forward_kernel (10000,16,9)x(16,16,1) **Time:** 60.29 msecond **Cycles:** 72,338,423 **Regs:** 32 **GPU:** TITAN V **SM Frequency:** 1.20 cycle/nsecond **CC:** 7.0 **Process:** [917] m3

GPU Speed Of Light

High-level overview of the utilization for compute and memory resources of the GPU.For each unit,the Speed Of Light (SOL) reports the achieved percentage of utilization with respect to the theoretical maximum.High-level overview of the utilization for compute and memory resources of the GPU presented as a roofline chart.

```
SOL SM [%]                 56.01 | Duration [msecond]                    60.29
SOL Memory [%]             81.42 | Elapsed Cycles [cycle]           72,338,423
SOL L1/TEX Cache [%]       81.41 | SM Active Cycles [cycle]      72,346,173.90
SOL L2 Cache [%]            9.96 | SM Frequency [cycle/nsecond]           1.20
SOL DRAM [%]               29.92 | DRAM Frequency [cycle/usecond]       850.38
```

SOL SM Breakdown		SOL Memory Breakdown	
SOL SM: Inst Executed Pipe Lsu [%]	56.01	SOL L1: Data Pipe Lsu Wavefronts [%]	81.42
SOL SM: Mio2rf Writeback Active [%]	32.08	SOL L1: Lsu Writeback Active [%]	67.69
SOL SM: Issue Active [%]	28.66	SOL L1: Lsuin Requests [%]	56.01
SOL SM: Inst Executed [%]	28.66	SOL GPU: Dram Throughput [%]	29.92
SOL SM: Mio Inst Issued [%]	28.25	SOL L1: Data Bank Reads [%]	10.27
SOL SM: Pipe Fma Cycles Active [%]	16.57	SOL L2: D Sectors Fill Device [%]	9.96
SOL SM: Pipe Alu Cycles Active [%]	11.48	SOL L2: T Sectors [%]	8.04
SOL SM: Inst Executed Pipe Xu [%]	1.99	SOL L2: Lts2xbar Cycles Active [%]	6.98
SOL SM: Inst Executed Pipe Adu [%]	0.96	SOL L2: Xbar2lts Cycles Active [%]	5.69
SOL SM: Inst Executed Pipe Cbu Pred On Any [%]	0.67	SOL L2: D Sectors [%]	4.77
SOL SM: Mio Pq Read Cycles Active [%]	0.16	SOL L2: T Tag Requests [%]	4.71
SOL SM: Mio Pq Write Cycles Active [%]	0.14	SOL L1: M Xbar2l1tex Read Sectors [%]	4.03
SOL IDC: Request Cycles Active [%]	0	SOL L1: M L1tex2xbar Req Cycles Active [%]	3.28
SOL SM: Inst Executed Pipe Tex [%]	0	SOL L1: Data Bank Writes [%]	0.55
SOL SM: Inst Executed Pipe Ipa [%]	0	SOL L1: F Wavefronts [%]	0.00
SOL SM: Inst Executed Pipe Fp16 [%]	0	SOL L1: Texin Sm2tex Req Cycles Active [%]	0.00
SOL SM: Pipe Fp64 Cycles Active [%]	0	SOL L2: D Sectors Fill Sysmem [%]	0
SOL SM: Pipe Shared Cycles Active [%]	0	SOL L1: Data Pipe Tex Wavefronts [%]	0
SOL SM: Pipe Tensor Cycles Active [%]	0	SOL L2: D Atomic Input Cycles Active [%]	0
		SOL L1: Tex Writeback Active [%]	0

The first conv_forward_kernel GPU utilization SM is 55.34%(compared to base line: 75.05%), the Memory is 91.09%(compared with base line: 88.46%)
The second conv_forward_kernel GPU utilization SM is 56.1%(compared to base line: 75.07%), the Memory is 81.42%(compared with base line: 78.90%)
The variable in loop unrolling will be placed in registers and increase the access speed. It cut down the utilization of SM but raise Memory, which increases the efficiency of execution.

e. What references did you use when implementing this technique?

   Restrict: https://docs.nvidia.com/cuda/cuda-c-programming-guide/index.html#restrict
   Loop unrolling: chapter 16

Optimization 3: Tiled shared memory convolution

a. Which optimization did you choose to implement and why did you choose that optimization technique.

   Tiled shared memory convolution (2 points)

b. How does the optimization work? Did you think the optimization would increase performance of the forward convolution? Why? Does the optimization synergize with any of your previous optimizations?

   The input data and weighted matrix are put in the shared memory. It is a very common technique used in convolution and can help to increase reuse rate and to reduce global memory access rate, in order to speed up the process. No synergy.

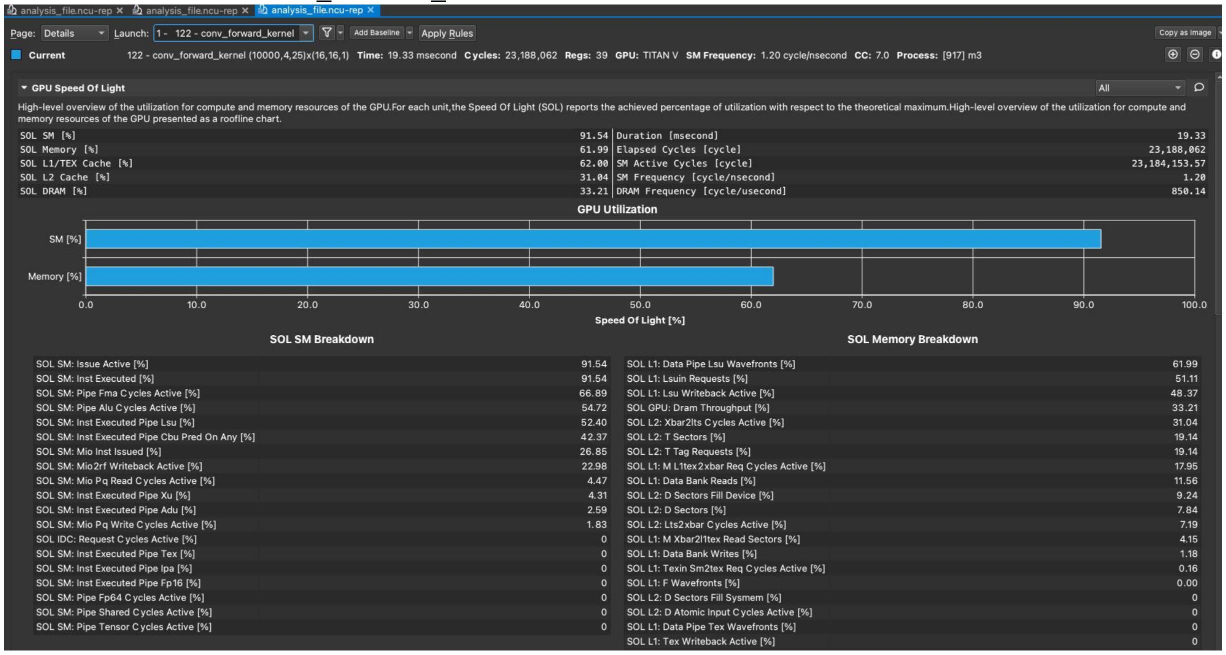
c. List the Op Times, whole program execution time, and accuracy for batch size of 100, 1k, and 10k images using this optimization (including any previous optimizations also used).

Batch Size	Op Time 1	Op Time 2	Total Execution Time	Accuracy
100	0.209385 ms	1.01973 ms	0m1.142s	0.86
1000	1.96323 ms	10.0182 ms	0m12.081s	0.886
10000	19.2785 ms	100.08 ms	1m42.191s	0.8714

d. Was implementing this optimization successful in improving performance? Why or why not? Include profiling results from nsys and Nsight-Compute to justify your answer, directly comparing to your baseline (or the previous optimization this one is built off of).

I think it's not successful to optimization.
profiling results from nsys and Nsight-Compute

Launch: 1 -   122 - conv_forward_kernel



Current   122 - conv_forward_kernel (10000,4,25)x(16,16,1) **Time:** 19.33 msecond **Cycles:** 23,188,062 **Regs:** 39 **GPU:** TITAN V **SM Frequency:** 1.20 cycle/nsecond **CC:** 7.0 **Process:** [917] m3

GPU Speed Of Light

High-level overview of the utilization for compute and memory resources of the GPU.For each unit,the Speed Of Light (SOL) reports the achieved percentage of utilization with respect to the theoretical maximum.High-level overview of the utilization for compute and memory resources of the GPU presented as a roofline chart.

SOL SM [%]	91.54	Duration [msecond]	19.33
SOL Memory [%]	61.99	Elapsed Cycles [cycle]	23,188,062
SOL L1/TEX Cache [%]	62.00	SM Active Cycles [cycle]	23,184,153.57
SOL L2 Cache [%]	31.04	SM Frequency [cycle/nsecond]	1.20
SOL DRAM [%]	33.21	DRAM Frequency [cycle/usecond]	850.14

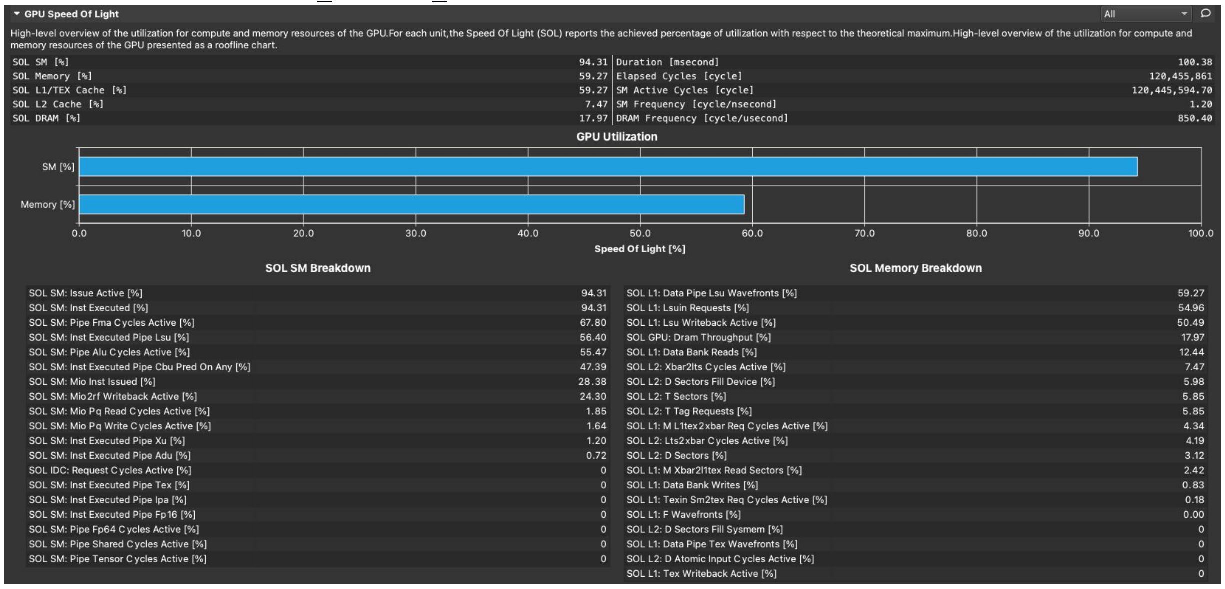
SOL SM Breakdown

SOL SM: Issue Active [%]	91.54
SOL SM: Inst Executed [%]	91.54
SOL SM: Pipe Fma Cycles Active [%]	66.89
SOL SM: Pipe Alu Cycles Active [%]	54.72
SOL SM: Inst Executed Pipe Lsu [%]	52.40
SOL SM: Inst Executed Pipe Cbu Pred On Any [%]	42.37
SOL SM: Mio Inst Issued [%]	26.85
SOL SM: Mio2rf Writeback Active [%]	22.98
SOL SM: Mio Pq Read Cycles Active [%]	4.47
SOL SM: Inst Executed Pipe Xu [%]	4.31
SOL SM: Inst Executed Pipe Adu [%]	2.59
SOL SM: Mio Pq Write Cycles Active [%]	1.83
SOL IDC: Request Cycles Active [%]	0
SOL SM: Inst Executed Pipe Tex [%]	0
SOL SM: Inst Executed Pipe Ipa [%]	0
SOL SM: Inst Executed Pipe Fp16 [%]	0
SOL SM: Pipe Fp64 Cycles Active [%]	0
SOL SM: Pipe Shared Cycles Active [%]	0
SOL SM: Pipe Tensor Cycles Active [%]	0

SOL Memory Breakdown

SOL L1: Data Pipe Lsu Wavefronts [%]	61.99
SOL L1: Lsuin Requests [%]	51.11
SOL L1: Lsu Writeback Active [%]	48.37
SOL GPU: Dram Throughput [%]	33.21
SOL L2: Xbar2lts Cycles Active [%]	31.04
SOL L2: T Sectors [%]	19.14
SOL L2: T Tag Requests [%]	19.14
SOL L1: M L1tex2xbar Req Cycles Active [%]	17.95
SOL L1: Data Bank Reads [%]	11.56
SOL L2: D Sectors Fill Device [%]	9.24
SOL L2: D Sectors [%]	7.84
SOL L2: Lts2xbar Cycles Active [%]	7.19
SOL L1: M Xbar2l1tex Read Sectors [%]	4.15
SOL L1: Data Bank Writes [%]	1.18
SOL L1: Texin Sm2tex Req Cycles Active [%]	0.16
SOL L1: F Wavefronts [%]	0.00
SOL L2: D Sectors Fill Sysmem [%]	0
SOL L2: D Atomic Input Cycles Active [%]	0
SOL L1: Data Pipe Tex Wavefronts [%]	0
SOL L1: Tex Writeback Active [%]	0

Launch: 4 -   143 - conv_forward_kernel



GPU Speed Of Light

High-level overview of the utilization for compute and memory resources of the GPU.For each unit,the Speed Of Light (SOL) reports the achieved percentage of utilization with respect to the theoretical maximum.High-level overview of the utilization for compute and memory resources of the GPU presented as a roofline chart.

SOL SM [%]	94.31	Duration [msecond]	100.38
SOL Memory [%]	59.27	Elapsed Cycles [cycle]	120,455,861
SOL L1/TEX Cache [%]	59.27	SM Active Cycles [cycle]	120,445,594.70
SOL L2 Cache [%]	7.47	SM Frequency [cycle/nsecond]	1.20
SOL DRAM [%]	17.97	DRAM Frequency [cycle/usecond]	850.40

SOL SM Breakdown

SOL SM: Issue Active [%]	94.31
SOL SM: Inst Executed [%]	94.31
SOL SM: Pipe Fma Cycles Active [%]	67.80
SOL SM: Inst Executed Pipe Lsu [%]	56.40
SOL SM: Pipe Alu Cycles Active [%]	55.47
SOL SM: Inst Executed Pipe Cbu Pred On Any [%]	47.39
SOL SM: Mio Inst Issued [%]	28.38
SOL SM: Mio2rf Writeback Active [%]	24.30
SOL SM: Mio Pq Read Cycles Active [%]	1.85
SOL SM: Mio Pq Write Cycles Active [%]	1.64
SOL SM: Inst Executed Pipe Xu [%]	1.20
SOL SM: Inst Executed Pipe Adu [%]	0.72
SOL IDC: Request Cycles Active [%]	0
SOL SM: Inst Executed Pipe Tex [%]	0
SOL SM: Inst Executed Pipe Ipa [%]	0
SOL SM: Inst Executed Pipe Fp16 [%]	0
SOL SM: Pipe Fp64 Cycles Active [%]	0
SOL SM: Pipe Shared Cycles Active [%]	0
SOL SM: Pipe Tensor Cycles Active [%]	0

SOL Memory Breakdown

SOL L1: Data Pipe Lsu Wavefronts [%]	59.27
SOL L1: Lsuin Requests [%]	54.96
SOL L1: Lsu Writeback Active [%]	50.49
SOL GPU: Dram Throughput [%]	17.97
SOL L1: Data Bank Reads [%]	12.44
SOL L2: Xbar2lts Cycles Active [%]	7.47
SOL L2: D Sectors Fill Device [%]	5.98
SOL L2: T Sectors [%]	5.85
SOL L2: T Tag Requests [%]	5.85
SOL L1: M L1tex2xbar Req Cycles Active [%]	4.34
SOL L2: Lts2xbar Cycles Active [%]	4.19
SOL L2: D Sectors [%]	3.12
SOL L1: M Xbar2l1tex Read Sectors [%]	2.42
SOL L1: Data Bank Writes [%]	0.83
SOL L1: Texin Sm2tex Req Cycles Active [%]	0.18
SOL L1: F Wavefronts [%]	0.00
SOL L2: D Sectors Fill Sysmem [%]	0
SOL L1: Data Pipe Tex Wavefronts [%]	0
SOL L2: D Atomic Input Cycles Active [%]	0
SOL L1: Tex Writeback Active [%]	0

The first conv_forward_kernel GPU utilization SM is 91.54%(compared to base line: 75.05%), the Memory is (compared with base line: 88.46%)

The second conv_forward_kernel GPU utilization SM is 94.31%(compared to base line: 75.07%), the Memory is 59.27%(compared with base line: 78.90%)
Not successful. Reason: many control divergence are introduced into this program, which slow down the running speed, although the using of shared memory can reduce global memory access.

e. What references did you use when implementing this technique?

   Tiled shared memory, chapter 7

Optimization 4: Shared memory matrix multiplication and input matrix unrolling

f. Which optimization did you choose to implement and why did you choose that optimization technique.

   Shared memory matrix multiplication and input matrix unrolling (3 points)

g. How does the optimization work? Did you think the optimization would increase performance of the forward convolution? Why? Does the optimization synergize with any of your previous optimizations?

Two kernels were launched one by another. One is about making input matrix to unroll, another is to do matrix multiplication. (The trick is on unrolling while matrix multiplication is just normal one) Unrolling can make it easier for GPU access.

No synergy.

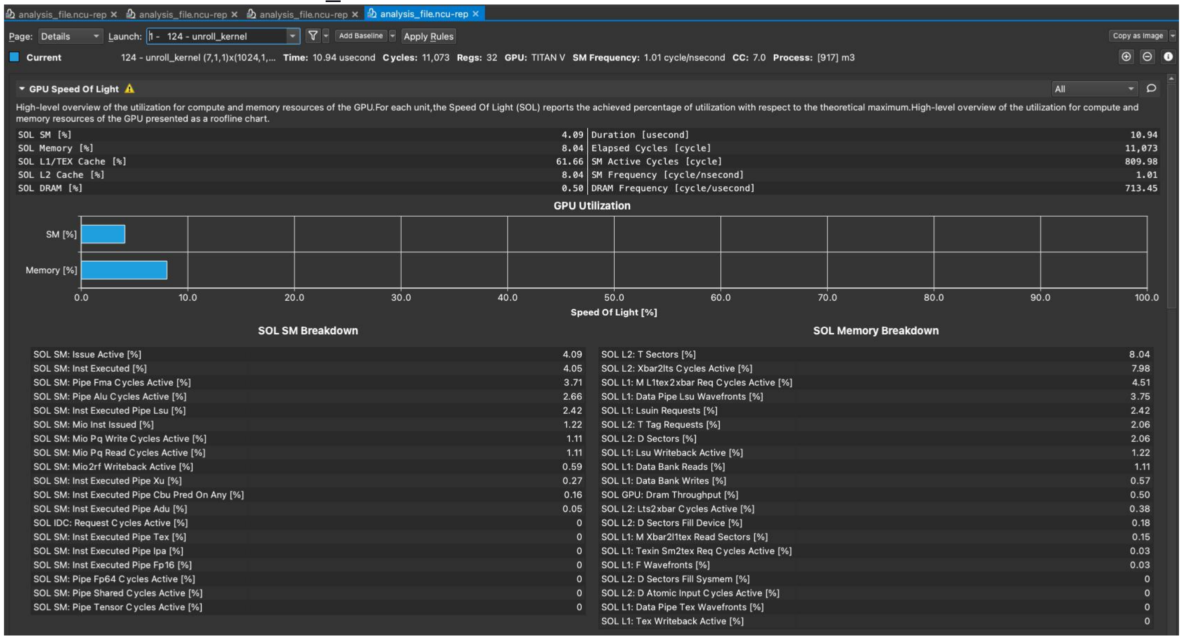
h. List the Op Times, whole program execution time, and accuracy for batch size of 100, 1k, and 10k images using this optimization (including any previous optimizations also used).

Batch Size	Op Time 1	Op Time 2	Total Execution Time	Accuracy
100	2.28242 ms	3.10461 ms	0m1.144s	0.86
1000	22.7806 ms	29.7191 ms	0m10.278s	0.886
10000	225.165 ms	296.176 ms	1m42.856s	0.8714

i. Was implementing this optimization successful in improving performance? Why or why not? Include profiling results from nsys and Nsight-Compute to justify your answer, directly comparing to your baseline (or the previous optimization this one is built off of).

I think it’s not successful to optimization.
profiling results from nsys and Nsight-Compute

Launch: 1 -   124 - unroll_kernel



Page: Details   Launch: 1 - 124 - unroll_kernel

Current   124 - unroll_kernel (7,1,1)x(1024,1,... **Time:** 10.94 usecond **Cycles:** 11,073 **Regs:** 32 **GPU:** TITAN V **SM Frequency:** 1.01 cycle/nsecond **CC:** 7.0 **Process:** [917] m3

GPU Speed Of Light

High-level overview of the utilization for compute and memory resources of the GPU.For each unit,the Speed Of Light (SOL) reports the achieved percentage of utilization with respect to the theoretical maximum.High-level overview of the utilization for compute and memory resources of the GPU presented as a roofline chart.

```
SOL SM [%]                 4.09 | Duration [usecond]               10.94
SOL Memory [%]             8.04 | Elapsed Cycles [cycle]          11,073
SOL L1/TEX Cache [%]      61.66 | SM Active Cycles [cycle]        809.98
SOL L2 Cache [%]           8.04 | SM Frequency [cycle/nsecond]      1.01
SOL DRAM [%]               0.50 | DRAM Frequency [cycle/usecond]  713.45
```

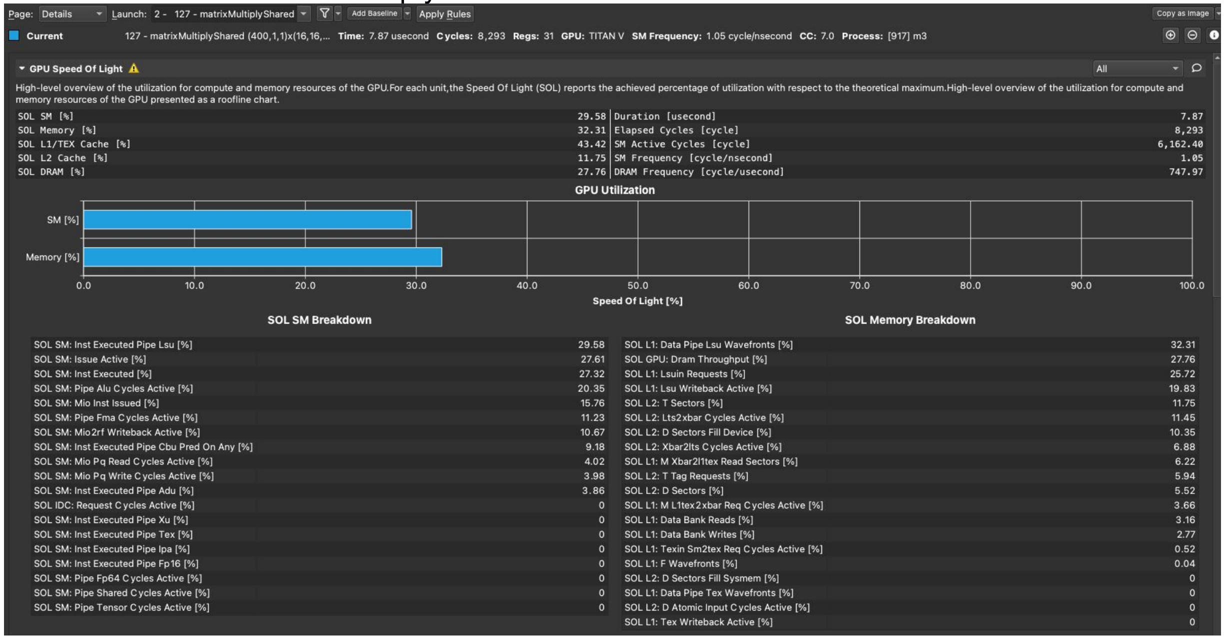
SOL SM Breakdown

SOL SM: Issue Active [%]	4.09
SOL SM: Inst Executed [%]	4.05
SOL SM: Pipe Fma Cycles Active [%]	3.71
SOL SM: Pipe Alu Cycles Active [%]	2.66
SOL SM: Inst Executed Pipe Lsu [%]	2.42
SOL SM: Mio Inst Issued [%]	1.22
SOL SM: Mio Pq Write Cycles Active [%]	1.11
SOL SM: Mio Pq Read Cycles Active [%]	1.11
SOL SM: Mio2rf Writeback Active [%]	0.59
SOL SM: Inst Executed Pipe Xu [%]	0.27
SOL SM: Inst Executed Pipe Cbu Pred On Any [%]	0.16
SOL SM: Inst Executed Pipe Adu [%]	0.05
SOL IDC: Request Cycles Active [%]	0
SOL SM: Inst Executed Pipe Tex [%]	0
SOL SM: Inst Executed Pipe Ipa [%]	0
SOL SM: Inst Executed Pipe Fp16 [%]	0
SOL SM: Pipe Fp64 Cycles Active [%]	0
SOL SM: Pipe Shared Cycles Active [%]	0
SOL SM: Pipe Tensor Cycles Active [%]	0

SOL Memory Breakdown

SOL L2: T Sectors [%]	8.04
SOL L2: Xbar2lts Cycles Active [%]	7.98
SOL L1: M L1tex2xbar Req Cycles Active [%]	4.51
SOL L1: Data Pipe Lsu Wavefronts [%]	3.75
SOL L1: Lsuin Requests [%]	2.42
SOL L2: T Tag Requests [%]	2.06
SOL L2: D Sectors [%]	2.06
SOL L1: Lsu Writeback Active [%]	1.22
SOL L1: Data Bank Reads [%]	1.11
SOL L1: Data Bank Writes [%]	0.57
SOL GPU: Dram Throughput [%]	0.50
SOL L2: Lts2xbar Cycles Active [%]	0.38
SOL L2: D Sectors Fill Device [%]	0.18
SOL L1: M Xbar2l1tex Read Sectors [%]	0.15
SOL L1: Texin Sm2tex Req Cycles Active [%]	0.03
SOL L1: F Wavefronts [%]	0.03
SOL L2: D Sectors Fill Sysmem [%]	0
SOL L2: D Atomic Input Cycles Active [%]	0
SOL L1: Data Pipe Tex Wavefronts [%]	0
SOL L1: Tex Writeback Active [%]	0

Launch: 2 -   127 - matrixMultiplyShared



Page: Details   Launch: 2 - 127 - matrixMultiplyShared

Current   127 - matrixMultiplyShared (400,1,1)x(16,16,... **Time:** 7.87 usecond **Cycles:** 8,293 **Regs:** 31 **GPU:** TITAN V **SM Frequency:** 1.05 cycle/nsecond **CC:** 7.0 **Process:** [917] m3

GPU Speed Of Light

High-level overview of the utilization for compute and memory resources of the GPU.For each unit,the Speed Of Light (SOL) reports the achieved percentage of utilization with respect to the theoretical maximum.High-level overview of the utilization for compute and memory resources of the GPU presented as a roofline chart.

```
SOL SM [%]                29.58 | Duration [usecond]                7.87
SOL Memory [%]            32.31 | Elapsed Cycles [cycle]           8,293
SOL L1/TEX Cache [%]      43.42 | SM Active Cycles [cycle]      6,162.40
SOL L2 Cache [%]          11.75 | SM Frequency [cycle/nsecond]      1.05
SOL DRAM [%]              27.76 | DRAM Frequency [cycle/usecond]  747.97
```

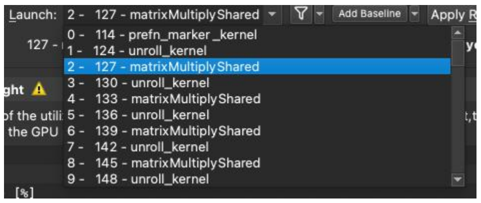
SOL SM Breakdown

SOL SM: Inst Executed Pipe Lsu [%]	29.58
SOL SM: Issue Active [%]	27.61
SOL SM: Inst Executed [%]	27.32
SOL SM: Pipe Alu Cycles Active [%]	20.35
SOL SM: Mio Inst Issued [%]	15.76
SOL SM: Pipe Fma Cycles Active [%]	11.23
SOL SM: Mio2rf Writeback Active [%]	10.67
SOL SM: Inst Executed Pipe Cbu Pred On Any [%]	9.18
SOL SM: Mio Pq Read Cycles Active [%]	4.02
SOL SM: Mio Pq Write Cycles Active [%]	3.98
SOL SM: Inst Executed Pipe Adu [%]	3.86
SOL IDC: Request Cycles Active [%]	0
SOL SM: Inst Executed Pipe Xu [%]	0
SOL SM: Inst Executed Pipe Tex [%]	0
SOL SM: Inst Executed Pipe Ipa [%]	0
SOL SM: Inst Executed Pipe Fp16 [%]	0
SOL SM: Pipe Fp64 Cycles Active [%]	0
SOL SM: Pipe Shared Cycles Active [%]	0
SOL SM: Pipe Tensor Cycles Active [%]	0

SOL Memory Breakdown

SOL L1: Data Pipe Lsu Wavefronts [%]	32.31
SOL GPU: Dram Throughput [%]	27.76
SOL L1: Lsuin Requests [%]	25.72
SOL L1: Lsu Writeback Active [%]	19.83
SOL L2: T Sectors [%]	11.75
SOL L2: Lts2xbar Cycles Active [%]	11.45
SOL L2: D Sectors Fill Device [%]	10.35
SOL L2: Xbar2lts Cycles Active [%]	6.88
SOL L1: M Xbar2l1tex Read Sectors [%]	6.22
SOL L2: T Tag Requests [%]	5.94
SOL L2: D Sectors [%]	5.52
SOL L1: M L1tex2xbar Req Cycles Active [%]	3.66
SOL L1: Data Bank Reads [%]	3.16
SOL L1: Data Bank Writes [%]	2.77
SOL L1: Texin Sm2tex Req Cycles Active [%]	0.52
SOL L1: F Wavefronts [%]	0.04
SOL L2: D Sectors Fill Sysmem [%]	0
SOL L1: Data Pipe Tex Wavefronts [%]	0
SOL L2: D Atomic Input Cycles Active [%]	0
SOL L1: Tex Writeback Active [%]	0

The first unroll_kernel GPU utilization SM is 4.09 % (compared to base line: 75.05%), the Memory is 8.04% (compared with base line: 88.46%)
The first matrixMultiplyShared GPU utilization SM is 29.58% (compared to base line: 75.07%), the Memory is 32.31% (compared with base line: 78.90%)



```
Launch: 2 - 127 - matrixMultiplyShared
0 - 114 - prefn_marker_kernel
1 - 124 - unroll_kernel
2 - 127 - matrixMultiplyShared
3 - 130 - unroll_kernel
4 - 133 - matrixMultiplyShared
5 - 136 - unroll_kernel
6 - 139 - matrixMultiplyShared
7 - 142 - unroll_kernel
8 - 145 - matrixMultiplyShared
9 - 148 - unroll_kernel
```

also there are many launches in kernels.

Not improve the performance(even worse a lot). The 1st reason is new X_unroll matrix is used, which leads to more layer time in order to allocate && copy space. And two separate kernels need more additional time to do memory data transfer. 2nd reason is unroll function needs more operations and no coalescing is used, so even with less time in forward function, the overall time is not better.

j. What references did you use when implementing this technique?

   Chapter 7, Chapter 16

Optimization 5: Kernel fusion for unrolling and matrix-multiplication

k. Which optimization did you choose to implement and why did you choose that optimization technique.

   Kernel fusion for unrolling and matrix-multiplication (requires previous optimization: Shared memory matrix multiplication and input matrix unrolling) (2 points)

l. How does the optimization work? Did you think the optimization would increase performance of the forward convolution? Why? Does the optimization synergize with any of your previous optimizations?

Does synergize with previous optimization: Shared memory matrix multiplication and input matrix unrolling.

This merges the unrolling kernel and matrix multiplication kernel into one kernel which can speed up the op time.(less duplicate work done for the kernel). And two shared memory are used, one for weighted matrix and another for input matrix.

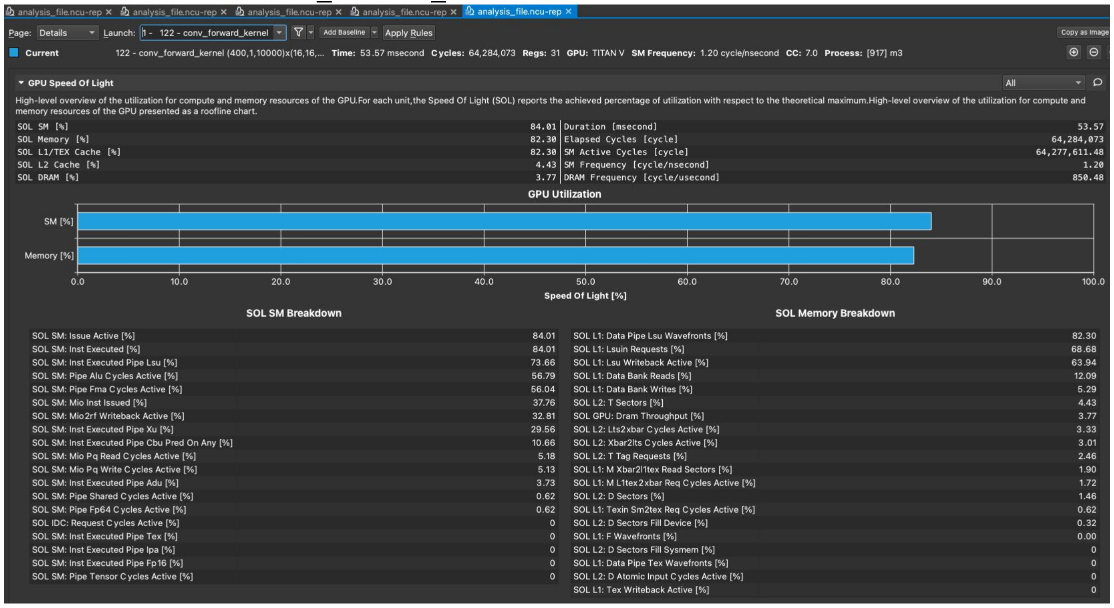
m. List the Op Times, whole program execution time, and accuracy for batch size of 100, 1k, and 10k images using this optimization (including any previous optimizations also used).

Batch Size	Op Time 1	Op Time 2	Total Execution Time	Accuracy
100	0.552482 ms	0.347056 ms	0m1.134s	0.86
1000	5.3891 ms	3.27264 ms	0m10.654s	0.886
10000	53.5837 ms	32.6165 ms	1m42.579s	0.8714

n. Was implementing this optimization successful in improving performance? Why or why not? Include profiling results from nsys and Nsight-Compute to justify your answer, directly comparing to your baseline (or the previous optimization this one is built off of).

I think it's not successful to optimization.
profiling results from nsys and Nsight-Compute.

Launch: 1 - 122 - conv_forward_kernel



Page: Details Launch: 1 - 122 - conv_forward_kernel

Current 122 - conv_forward_kernel (400,1,10000)x(16,16,... Time: 53.57 msecond Cycles: 64,284,073 Regs: 31 GPU: TITAN V SM Frequency: 1.20 cycle/nsecond CC: 7.0 Process: [917] m3

GPU Speed Of Light

High-level overview of the utilization for compute and memory resources of the GPU.For each unit,the Speed Of Light (SOL) reports the achieved percentage of utilization with respect to the theoretical maximum.High-level overview of the utilization for compute and memory resources of the GPU presented as a roofline chart.

```
SOL SM [%]              84.01 | Duration [msecond]                   53.57
SOL Memory [%]          82.30 | Elapsed Cycles [cycle]          64,284,073
SOL L1/TEX Cache [%]    82.30 | SM Active Cycles [cycle]     64,277,611.48
SOL L2 Cache [%]         4.43 | SM Frequency [cycle/nsecond]          1.20
SOL DRAM [%]             3.77 | DRAM Frequency [cycle/usecond]      850.48
```

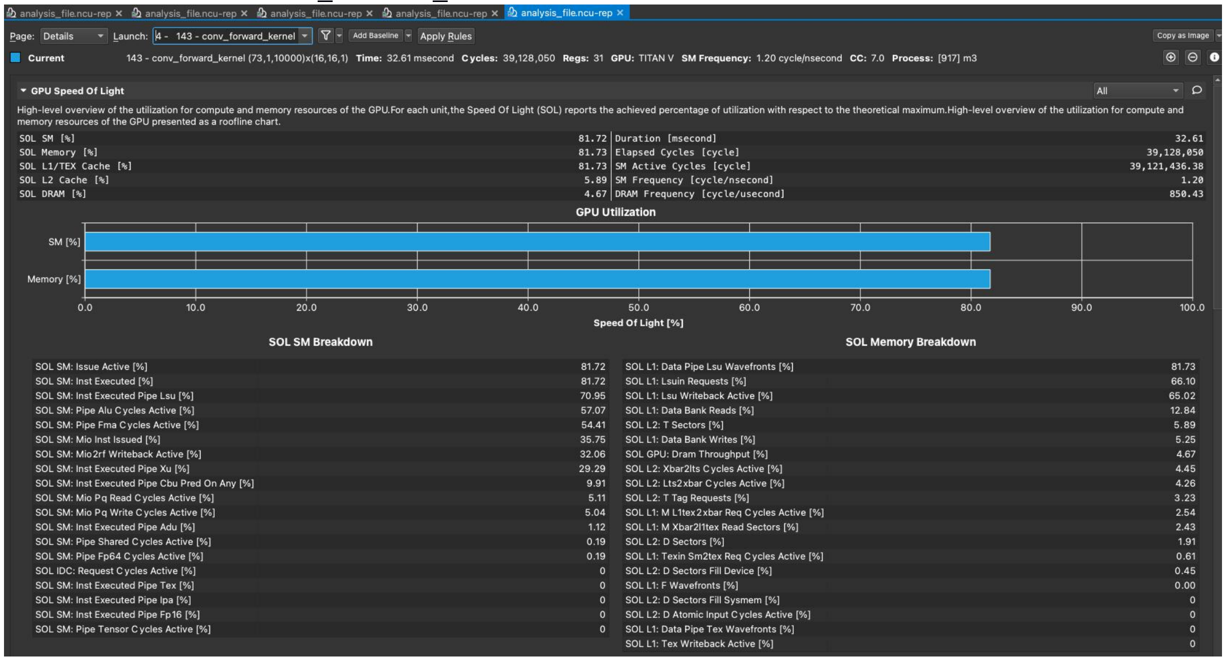
SOL SM Breakdown

SOL SM: Issue Active [%]	84.01
SOL SM: Inst Executed [%]	84.01
SOL SM: Inst Executed Pipe Lsu [%]	73.66
SOL SM: Pipe Alu Cycles Active [%]	56.79
SOL SM: Pipe Fma Cycles Active [%]	56.04
SOL SM: Mio Inst Issued [%]	37.76
SOL SM: Mio2rf Writeback Active [%]	32.81
SOL SM: Inst Executed Pipe Xu [%]	29.56
SOL SM: Inst Executed Pipe Cbu Pred On Any [%]	10.66
SOL SM: Mio Pq Read Cycles Active [%]	5.18
SOL SM: Mio Pq Write Cycles Active [%]	5.13
SOL SM: Inst Executed Pipe Adu [%]	3.73
SOL SM: Pipe Shared Cycles Active [%]	0.62
SOL SM: Pipe Fp64 Cycles Active [%]	0.62
SOL IDC: Request Cycles Active [%]	0
SOL SM: Inst Executed Pipe Tex [%]	0
SOL SM: Inst Executed Pipe Ipa [%]	0
SOL SM: Inst Executed Pipe Fp16 [%]	0
SOL SM: Pipe Tensor Cycles Active [%]	0

SOL Memory Breakdown

SOL L1: Data Pipe Lsu Wavefronts [%]	82.30
SOL L1: Lsuin Requests [%]	68.68
SOL L1: Lsu Writeback Active [%]	63.94
SOL L1: Data Bank Reads [%]	12.09
SOL L1: Data Bank Writes [%]	5.29
SOL L2: T Sectors [%]	4.43
SOL GPU: Dram Throughput [%]	3.77
SOL L2: Lts2xbar Cycles Active [%]	3.33
SOL L2: Xbar2lts Cycles Active [%]	3.01
SOL L2: T Tag Requests [%]	2.46
SOL L1: M Xbar2l1tex Read Sectors [%]	1.90
SOL L1: M L1tex2xbar Req Cycles Active [%]	1.72
SOL L2: D Sectors [%]	1.46
SOL L1: Texin Sm2tex Req Cycles Active [%]	0.62
SOL L2: D Sectors Fill Device [%]	0.32
SOL L1: F Wavefronts [%]	0.00
SOL L2: D Sectors Fill Sysmem [%]	0
SOL L1: Data Pipe Tex Wavefronts [%]	0
SOL L2: D Atomic Input Cycles Active [%]	0
SOL L1: Tex Writeback Active [%]	0

Launch: 4 -   143 - conv_forward_kernel



Page: Details Launch: 4 - 143 - conv_forward_kernel

Current 143 - conv_forward_kernel (73,1,10000)x(16,16,1) Time: 32.61 msecond Cycles: 39,128,050 Regs: 31 GPU: TITAN V SM Frequency: 1.20 cycle/nsecond CC: 7.0 Process: [917] m3

GPU Speed Of Light

High-level overview of the utilization for compute and memory resources of the GPU.For each unit,the Speed Of Light (SOL) reports the achieved percentage of utilization with respect to the theoretical maximum.High-level overview of the utilization for compute and memory resources of the GPU presented as a roofline chart.

```
SOL SM [%]                 81.72 | Duration [msecond]                  32.61
SOL Memory [%]             81.73 | Elapsed Cycles [cycle]         39,128,050
SOL L1/TEX Cache [%]       81.73 | SM Active Cycles [cycle]    39,121,436.38
SOL L2 Cache [%]            5.89 | SM Frequency [cycle/nsecond]         1.20
SOL DRAM [%]                4.67 | DRAM Frequency [cycle/usecond]     850.43
```

SOL SM Breakdown

SOL SM: Issue Active [%]	81.72
SOL SM: Inst Executed [%]	81.72
SOL SM: Inst Executed Pipe Lsu [%]	70.95
SOL SM: Pipe Alu Cycles Active [%]	57.07
SOL SM: Pipe Fma Cycles Active [%]	54.41
SOL SM: Mio Inst Issued [%]	35.75
SOL SM: Mio2rf Writeback Active [%]	32.06
SOL SM: Inst Executed Pipe Xu [%]	29.29
SOL SM: Inst Executed Pipe Cbu Pred On Any [%]	9.91
SOL SM: Mio Pq Read Cycles Active [%]	5.11
SOL SM: Mio Pq Write Cycles Active [%]	5.04
SOL SM: Inst Executed Pipe Adu [%]	1.12
SOL SM: Pipe Shared Cycles Active [%]	0.19
SOL SM: Pipe Fp64 Cycles Active [%]	0.19
SOL IDC: Request Cycles Active [%]	0
SOL SM: Inst Executed Pipe Tex [%]	0
SOL SM: Inst Executed Pipe Ipa [%]	0
SOL SM: Inst Executed Pipe Fp16 [%]	0
SOL SM: Pipe Tensor Cycles Active [%]	0

SOL Memory Breakdown

SOL L1: Data Pipe Lsu Wavefronts [%]	81.73
SOL L1: Lsuin Requests [%]	66.10
SOL L1: Lsu Writeback Active [%]	65.02
SOL L1: Data Bank Reads [%]	12.84
SOL L2: T Sectors [%]	5.89
SOL L1: Data Bank Writes [%]	5.25
SOL GPU: Dram Throughput [%]	4.67
SOL L2: Xbar2lts Cycles Active [%]	4.45
SOL L2: Lts2xbar Cycles Active [%]	4.26
SOL L2: T Tag Requests [%]	3.23
SOL L1: M L1tex2xbar Req Cycles Active [%]	2.54
SOL L1: M Xbar2l1tex Read Sectors [%]	2.43
SOL L2: D Sectors [%]	1.91
SOL L1: Texin Sm2tex Req Cycles Active [%]	0.61
SOL L2: D Sectors Fill Device [%]	0.45
SOL L1: F Wavefronts [%]	0.00
SOL L2: D Sectors Fill Sysmem [%]	0
SOL L2: D Atomic Input Cycles Active [%]	0
SOL L1: Data Pipe Tex Wavefronts [%]	0
SOL L1: Tex Writeback Active [%]	0

The first conv_forward_kernel GPU utilization SM is 84.01% (compared to base line: 75.05%), the Memory is 82.30% (compared with base line: 88.46%)
The second conv_forward_kernel GPU utilization SM is 81.72% (compared to base line: 75.07%), the Memory is 81.73% (compared with base line: 78.90%)
This method does perform better than previous two kernel version(without kernel fusing), though it isn't as good as the base line version(const-weight). Too many math operations are introduced during the unroll process can slow down the execution.

o. What references did you use when implementing this technique?

Chapter 7, Chapter 16