گام ۱)

زمان اجرا این گام برای n = 32 به صورت زیر می باشد.

```
[-] N = 32
MatrixA(32,32), MatrixB(32,32)
Computing result using CUDA Kernel...
Elapsed time in msec = 0.028704
```

همچنین خروجی به ازای n = 4 به صورت زیر می باشد.

```
[-] N = 4
MatrixA(4,4), MatrixB(4,4)
Computing result using CUDA Kernel...
Elapsed time in msec = 0.018464
[-] Matrix C
0.040000 0.040000 0.040000 0.040000
0.030000 0.030000 0.030000 0.030000
0.040000 0.040000 0.040000 0.040000
0.030000 0.030000 0.030000 0.030000
```

گام ۲)

نتایج خروجی به ازای n = 1024 برای روش های اول تا سوم به صورت زیر می باشد.

روش اول)

در این روش یک بلوک داریم و درون آن بلوک ۳۲ * ۳۲ یعنی ۱۰۲۴ نخ قرار دارند. اندازه کاشی را برابر با ۳۲ در نظر گرفته ایم. چون با توجه به اینکه هر نخ باید کاری بیشتری انجام دهد و ۱۰۲۴ نخ بیشتر نداریم باید به هر نخ یک مربع ۳۲ در ۳۲ بدهیم. در نتیجه این کار تمام مربع ۱۰۲۴ در ۱۰۲۴ به نخ ها مختلف می رسد.

نتیجه به صورت زیر می باشد.

```
[Matrix Multiply Using CUDA] - Starting...
GPU Device 0: "Tesla T4" with compute capability 7.5

[-] N = 1024
MatrixA(1024,1024), MatrixB(1024,1024)
Computing result using CUDA Kernel...
Elapsed time in msec = 1023.028564
```

روش دوم)

در این حالت کد داخل کرنل شبیه به گام اول است و تنها تفاوت افزایش تعداد بلاک هاست. با توجه به اینکه ابعاد ماتریس ۱۰۲۴ در ۱۰۲۴ می باشد و هر بلاک با توجه به محدودیت ۱۰۲۴ تایی روی نخ ها توانایی انجام کار برای یک مربع ۳۲ در ۳۲ را دارد، نیاز به ۱۰۲۴/۳۲ = ۳۲ ضربدر ۳۲ بلاک خواهیم داشت. یعنی ورودی مان برای شکل گرید به صورت (32, 32, 1) است.

نتیجه به صورت زیر می باشد.

```
[Matrix Multiply Using CUDA] - Starting...
GPU Device 0: "Tesla T4" with compute capability 7.5

[-] N = 1024
MatrixA(1024,1024), MatrixB(1024,1024)
Computing result using CUDA Kernel...
Elapsed time in msec = 6.490240
```

روش سوم)

در این حالت که ترکیب دو روش بالا است ابتدا ماتریس ۱۰۲۴ در ۱۰۲۴ را به بلاک های با ابعاد ۱۲۸ در ۱۲۸ می شکنیم. (یعنی ۸ * ۸ = ۶۴ بلاک خواهیم داشت). سپس درون بلاک ها با اندازه کاشی کاری برابر با ۴، کار هر مربع ۴ در ۴ را به یک نخ می دهیم. چون ۱۲۸/۴ = ۳۲ پس تعداد نخ ها نیز دقیقا همان ۱۰۲۴ خواهد بود.

نتیجه به صورت زیر می باشد.

```
[Matrix Multiply Using CUDA] - Starting...
GPU Device 0: "Tesla T4" with compute capability 7.5

[-] N = 1024
MatrixA(1024,1024), MatrixB(1024,1024)
Computing result using CUDA Kernel...
Elapsed time in msec = 22.213120
```

همان طور که دیده می شود نتایج در حالت دوم که تعداد بلاک ها بیشتر از دیگر حالت ها می باشد بهتر است. پس از آن حالت سوم است که تعداد بلاک ها در آن متوسط است و در داخل آن نیز کاشی کاره کرده ایم بنابراین کار نخ ها زیاد تر از حالت دوم است. در انتها و بدترین حالت نیز حالت اول می باشد که در آن فقط یک بلاک داریم و تمامی کار های مربوط به ماتریس بزرگ بین ۱۰۲۴ نخ موجود در این بلاک تقسیم می شود.

گام ۳)

در این حالت اندازه کاشی ۳۲ در نظر گرفته شده است. و از حافظه مشترک که برای تمامی نخ های موجود در هر کاشی کاربرد دارد استفاده شده است. تعداد بلاک هایمان به اندازه ۳۲/۱۰۲۴ = ۳۲ ضربدر ۳۲ می باشد. درون هر بلاک نیز به تعداد ۳۲ در ۳۲ = ۱۰۲۴ نخ داریم که برابر ماکسیمم مقدار ممکن است.

نتیجه به صورت زیر می باشد.

```
[Matrix Multiply Using CUDA] - Starting...
GPU Device 0: "Tesla T4" with compute capability 7.5

[-] N = 1024
MatrixA(1024,1024), MatrixB(1024,1024)
Computing result using CUDA Kernel...
Elapsed time in msec = 5.279136
```

همان طور که مشخص است در این حالت جواب بسیار بهتر از حالت قبلی کاشی کاری می باشد. همچنین نسبت به حالت دوم گام ۲ نیز نتایج تا حدی بهبود یافته است.