#include<stdio.h>

#include<cuda.h>

#define row1 2 /\* Number of rows of first matrix \*/

#define col1 3 /\* Number of columns of first matrix \*/

#define row2 3 /\* Number of rows of second matrix \*/

#define col2 2 /\* Number of columns of second matrix \*/

\_\_global\_\_ void matproduct(int \*l,int \*m, int \*n)

{

int x=blockIdx.x;

int y=blockIdx.y;

int k;

n[col2\*y+x]=0;

for(k=0;k<col1;k++)

{

n[col2\*y+x]=n[col2\*y+x]+l[col1\*y+k]\*m[col2\*k+x];

}

}

int main()

{

int a[row1][col1];

int b[row2][col2];

int c[row1][col2];

int \*d,\*e,\*f;

int i,j;

printf("\n Enter elements of first matrix of size 2\*3\n");

for(i=0;i<row1;i++)

{

for(j=0;j<col1;j++)

{

scanf("%d",&a[i][j]);

}

}

printf("\n Enter elements of second matrix of size 3\*2\n");

for(i=0;i<row2;i++)

{

for(j=0;j<col2;j++)

{

scanf("%d",&b[i][j]);

}

}

cudaMalloc((void \*\*)&d,row1\*col1\*sizeof(int));

cudaMalloc((void \*\*)&e,row2\*col2\*sizeof(int));

cudaMalloc((void \*\*)&f,row1\*col2\*sizeof(int));

cudaMemcpy(d,a,row1\*col1\*sizeof(int),cudaMemcpyHostToDevice);

cudaMemcpy(e,b,row2\*col2\*sizeof(int),cudaMemcpyHostToDevice);

dim3 grid(col2,row1);

matproduct<<<grid,1>>>(d,e,f);

cudaMemcpy(c,f,row1\*col2\*sizeof(int),cudaMemcpyDeviceToHost);

printf("\nProduct of two matrices:\n ");

for(i=0;i<row1;i++)

{

for(j=0;j<col2;j++)

{

printf("%d\t",c[i][j]);

}

printf("\n");

}

cudaFree(d);

cudaFree(e);

cudaFree(f);

return 0;

}

Output:

Enter elements of first matrix of size 2\*3

1 2 3 4 5 6

Enter elements of second matrix of size 3\*2

7 8 9 10 11 12

Product of two matrices:

58 64

139 154