%%cu

#include <stdio.h>

#include<cuda.h>

#include <stdlib.h>

#include<time.h>

\_\_global\_\_ void vectsum(int \*x,int \*y,int \*z,int SIZE)

{

int tid=blockIdx.x\*blockDim.x+threadIdx.x;

if(tid<SIZE) {

z[tid]=x[tid]+y[tid];

}

}

int main(void)

{

int i;

int SIZE=100;

srand(time(NULL));

int a[SIZE],b[SIZE],c[SIZE];

int \*dev\_a,\*dev\_b,\*dev\_c;

cudaMalloc(&dev\_a, SIZE\*sizeof(int));

cudaMalloc(&dev\_b, SIZE\*sizeof(int));

cudaMalloc(&dev\_c, SIZE\*sizeof(int));

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

{

a[i] = rand()%10+1;

}

printf("\nThe 1st vector is:\n");

for(i=0;i<5;i++)

{

printf("%d ",a[i]);

}

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

{

b[i] = rand()%20+1;

}

printf("\nThe 2nd vector is:\n");

for(i=0;i<5;i++)

{

printf("%d ",b[i]);

}

cudaMemcpy(dev\_a,a,sizeof(a),cudaMemcpyHostToDevice);

cudaMemcpy(dev\_b,b,sizeof(b),cudaMemcpyHostToDevice);

vectsum<<<4,SIZE/4>>>(dev\_a,dev\_b,dev\_c,SIZE);

cudaMemcpy(&c,dev\_c,sizeof(c),cudaMemcpyDeviceToHost);

printf("\nThe result is:\n");

for(int i=0;i<SIZE;i++)

{

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

}

return 0;

}