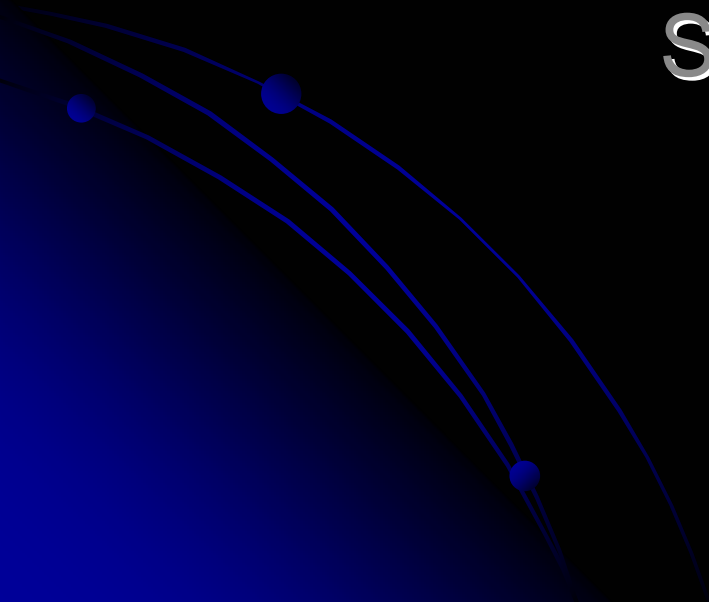


# GPU [1]

Speaker 高崇閔



# Exceed limitation

## Exercise2

```
// test C = A + B
void runTest(int argc, char** argv)
{
    unsigned int N = 1024 ;
    printf("N = %d\n", N);

    CUT_DEVICE_INIT(argc, argv);

    // set seed for rand()
    srand(2006);

    // allocate host memory for matrices A and B
    unsigned int size_A = N ;
    unsigned int mem_size_A = sizeof(float) * size_A ;
    float* h_A = (float*) malloc(mem_size_A);

    unsigned int size_B = N ;
    unsigned int mem_size_B = sizeof(float) * size_B ;
    float* h_B = (float*) malloc(mem_size_B);
```

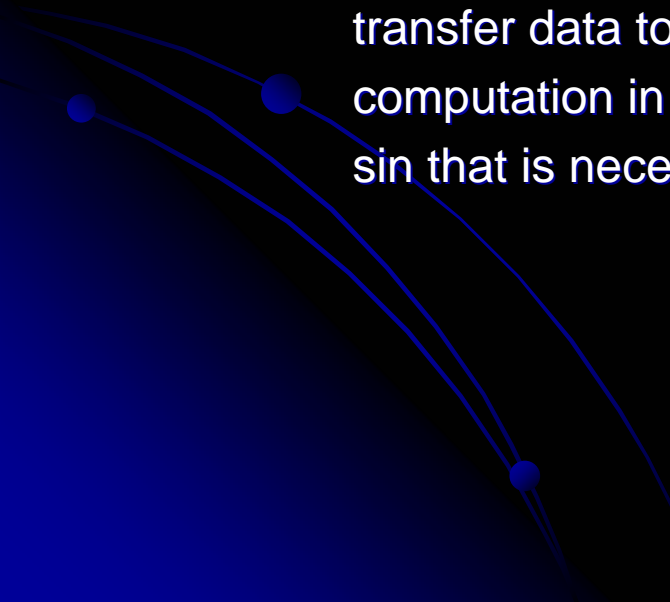
Exceed limitation

Error message

```
C:\Windows\system32\cmd.exe
N = 1024
Using device 0: GeForce 8800 GT
in GPU, C = A + B: 0.061740 (ms)
Cuda error: Kernel execution failed in file '.\vecadd_GPU.cu' in line 47 : invalid configuration argument.
請按任意鍵繼續 . . .
```

Line  
47

# CPU V.S. GPU

- Question : If we can speed up the computation of CPU, it's no use about GPU, doesn't it ?
  - Reply : In Table II, we spend twice time to transfer data than computation. But, it's not means we can take place of GPU by CPU. We just transfer data to GPU once, but we can do several times computation in GPU . As a result, the time we cost in transfer is sin that is necessities.
- 

Thread=512    N=number of block \* threads    size= size of (float) byte

experimental platform : Geforce 8800 GT

Exercise4

Table II

Num of block	size		GPU	Drive -> Host	CPU
16	32	KB	1.194096	0.126273	0
32	64	KB	1.228648	0.181308	0
64	128	KB	1.217473	0.378819	0
128	256	KB	1.250997	0.498387	0
256	512	KB	1.295416	0.999848	0
512	1.024	MB	1.357435	1.561372	0
1024	2.048	MB	1.463314	2.815442	0
2048	4.096	MB	1.780115	4.651988	16
4096	8.192	MB	2.215086	9.067375	15
8192	1.6384	MB	3.217448	18.717743	31
16384	3.2768	MB	5.030807	38.326660	63
32768	6.5536	MB	8.637410	65.668404	125
65536	131	MB	15.667075	135.295959	266

# Computation

## Matrix multiple

- $A : m \times n$ ,  $B : n \times p$ .  $A * B : m \times p$

The data we need to transfer is  $n \times (p+m) \times \text{sizeof(float)} \text{ byte}$

The times we do computation is  $(m \times p) \times (\text{addition}) + (n^2) \times (\text{plus})$

## Vector addition

- $A : 1 \times n$ ,  $B : 1 \times n$ .  $A + B : 1 \times n$

The data we need to transfer is  $(2 \times n) \times \text{sizeof(float)} \text{ byte}$

The times we do computation is  $(n) \times (\text{addition})$

## Ratio of addition and multiple (*multiple / addition*)

The ratio of data transfer is  $(p+m)$

The ratio of computation is  $[(m \times p) \times (\text{addition}) + (n^2) \times (\text{plus})] / (n) \times (\text{addition})$

# remain

- How to do several data in finite thread
- How to computation (multiple) between matrix and vector (inner product and outer product)

