Ex01

Authors:

- Moritz Gutfleisch
- Tuoxing Liu

Configuration

If you want to configure the behavior of the program, change the corresponding MACRO.

```
#define IS_COPYKERNEL 0 // 0: original, 1: using copyKernel
#define COPY_SIZE 1     // possible value: 1, 4, 8, 16
#define INSTR_PER_THREAD 4 //suggestion: 1, 2, 4, 8
```

Then compile again.

Task 1.1 results

copy type	Host to Device Bandwidth	Device to Host Bandwidth	Device to Device Bandwidth
cudaMemcpy	12.5	13	313.7
cudaMemcpy	12.7	13	313.8
cudaMemcpy	12.7	13	313.5
Average	12.63333	13	313.6667

copy type	Host to Device Bandwidth	Device to Host Bandwidth	Device to Device Bandwidth
copyKernel	12.3	11.8	166.7
copyKernel	12.3	12	164.8
copyKernel	12.3	11.9	165
Average	12.3	11.9	165.5

As it shows in the tables, the difference between `cudaMemcpy` and `copyKernel` is relatively small in H2D, and D2H. `copyKernel` might be slightly slower than `cudaMemcpy` in these 2 cases. However, in D2D situation, `copyKernel` is significantly (about 47%) slower than `cudaMemcpy`.

Task 1.2 results

copy size	number of transfer instructions per thread	Host to Device Bandwidth	Device to Host Bandwidth	Device to Device Bandwidth	Transfer Size (Bytes)
4	1	12.4	8.2	279	100000000
4	2	12.5	8.4	263.8	100000000
4	4	11.6	8.1	265.8	100000000
4	8	9.3	7.8	276.2	100000000
8	1	12.5	8.5	259.2	100000000
8	**2**	**10.5**	**8.1**	**288**	100000000
8	4	9.5	8	251.1	100000000
8	8	8.9	7.7	255.9	100000000
16	1	9.8	8.3	246.8	100000000
16	2	9.4	8.1	271.3	100000000
16	4	9.1	7.8	277	100000000
16	8	8.7	7.6	268.9	100000000

The result of D2D bandwidth is unstable. In this case, `copy size = 8, number of transfer instrunctions per thread = 2` obtained the best performance.

.