HPC - HW 2

Kitty Li (wl2407)

Processor: Intel(R) Core(TM) i9-9980HK CPU @ 2.40GHz
Github repository: https://github.com/lwyhasacat/HPCHW2

Problem 1.

See github page for modified code.

Problem 2.

Dimension	Time	Gflop/s	GB/s	Error
16	0.879390	2.274307	36.388916	1.047738e-09
64	0.775019	2.580787	41.292587	3.776222e-09
112	0.909909	2.198701	35.179210	7.275958e-10
160	0.764550	2.625126	42.002014	1.236913e-10
208	0.807255	2.497050	39.952804	4.638423e-11
256	0.787463	2.556648	40.906372	1.637090e-11
304	0.760723	2.659051	42.544819	1.045919e-11
352	0.753604	2.662212	42.595391	7.730705e-12
400	0.780408	2.624268	41.988293	3.865352e-12
448	0.810350	2.663009	42.608145	2.046363e-12
496	0.831522	2.641458	42.263336	1.591616e-12
544	0.852260	2.644555	42.312882	1.136868e-12
592	0.830742	2.497462	39.959398	6.821210e-13
640	0.841313	2.492713	39.883411	5.684342e-13
688	1.034977	2.517240	40.275838	5.684342e-13
736	0.913807	2.617762	41.884197	4.547474e-13
784	1.090622	2.651094	42.417510	4.547474e-13
832	0.913604	2.521576	40.345208	2.842171e-13
880	1.046164	2.605603	41.689647	2.842171e-13
928	1.234996	2.588442	41.415065	2.842171e-13
976	1.369411	2.715661	43.450584	3.410605e-13
1024	0.806683	2.662116	42.593855	1.705303e-13
1072	0.956234	2.576619	41.225901	1.705303e-13
1120	1.046387	2.685293	42.964693	1.705303e-13
1168	1.184452	2.690550	43.048799	2.273737e-13
1216	1.369003	2.626796	42.028734	2.273737e-13
1264	1.595618	2.531292	40.500676	2.273737e-13
1312	1.760182	2.566102	41.057633	2.273737e-13
1360	1.908219	2.636444	42.183100	2.273737e-13
1408	2.163076	2.580870	41.293925	2.842171e-13
1456	2.410105	2.561404	40.982471	2.273737e-13
1504	2.638942	2.578361	41.253770	2.273737e-13
1552	2.877914	2.597929	41.566869	2.842171e-13
1600	3.156445	2.595325	41.525197	2.842171e-13
1648	3.567700	2.509073	40.145167	3.410605e-13
1696	3.908536	2.496281	39.940492	2.842171e-13
1744	4.299293	2.467587	39.481385	2.842171e-13
1792	4.567392	2.519856	40.317696	2.842171e-13
1840	4.985559	2.499019	39.984308	2.842171e-13
1888	5.440342	2.474057	39.584918	3.410605e-13
1936	5.813841	2.496220	39.939524	3.410605e-13
1984	6.238334	2.503724	40.059577	3.410605e-13

Problem 3.

After modifying the code:

```
Reference time: 0.0038
Taylor time:    0.5700      Error: 6.927903e-12
Intrin time:    0.0020      Error: 6.927903e-12
Vector time:    0.0024      Error: 6.927903e-12
```

Problem 4b).

1)

```
0.923547 seconds
3.047804 cycles/eval
2.165491 Gflop/s
```

Figure 1: add

```
3.334147 seconds
11.002999 cycles/eval
0.599836 Gflop/s
```

Figure 2: division

```
3.397521 seconds
11.211955 cycles/eval
0.588657 Gflop/s
```

Figure 3: square

```
13.225354 seconds
43.643803 cycles/eval
0.151224 Gflop/s
```

Figure 4: sin

```
14.619524 seconds
48.244604 cycles/eval
0.136803 Gflop/s
```

Figure 5: cos

Run with this command: g++ -fopenmp -std=c++11 -O3 -march=native compute.cpp && ./a.out -n 1000000000

2).

```
time = 0.993726
flop-rate = 8.049982 Gflop/s

time = 1.055121
flop-rate = 7.581997 Gflop/s

time = 1.163457
flop-rate = 6.875912 Gflop/s
```

Figure 6: compute-vec report

The first one is slower than the second and the third, since the latter two use vectorization. The difference may not seem that obvious, but in my understanding it's because the VEC_LEN is only 4. If the vector size is larger, the difference will be more clear.

3).

```
time = 0.975893
flop-rate = 8.197250 Gflop/s

time = 0.989584
flop-rate = 8.084123 Gflop/s

time = 0.971313
flop-rate = 8.236181 Gflop/s
```

Figure 7: M = 1

```
time = 1.437741
flop-rate = 55.641142 Gflop/s

time = 1.986460
flop-rate = 40.272443 Gflop/s

time = 2.021559
flop-rate = 39.573183 Gflop/s
```

Figure 8: M = 10

```
time = 34.455333
flop-rate = 23.218435 Gflop/s

time = 33.160492
flop-rate = 24.125081 Gflop/s

time = 31.776848
flop-rate = 25.175555 Gflop/s
```

Figure 9: M = 100

I used M = 1, 10, and 100. It seems like for M small, we don't need to use parallel anyway and the pipelines "cannot be filled." When M = 1, the performance is much better because the parallel is actually working. When M is too large, data get overspilled and therefore the performance is not improved.