ACA PROJECT REPORT

THE TOMASULO ALGORITHM

Yash Thadhani

ythadhani3@gatech.edu

Validation of Traces provided:

1] gcc.100k.trace

As seen in the screenshot my outputs tally with the outputs provided for the default configuration:

```
Processor Settings
R: 2
k0: 3
k1: 2
k2: 1
F: 4

Processor stats:
Total instructions: 100000
Avg Dispatch queue size: 26039.072266
Maximum Dispatch queue size: 51965
Avg inst fired per cycle: 1.921303
Avg inst retired per cycle: 1.921303
Total run time (cycles): 52048
```

2] gobmk.100k.trace

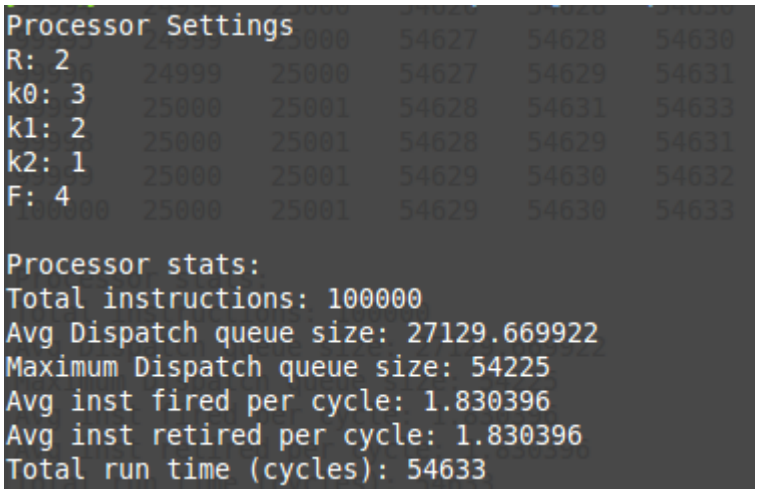
As seen in the screenshot my outputs tally with the outputs provided for the default configuration:

```
Processor Settings
R: 2
k0: 3
k1: 2
k2: 1
F: 4

Processor stats:
Total instructions: 100000
Avg Dispatch queue size: 27406.449219
Maximum Dispatch queue size: 55374
Avg inst fired per cycle: 1.828421
Avg inst retired per cycle: 1.828421
Total run time (cycles): 54692
```

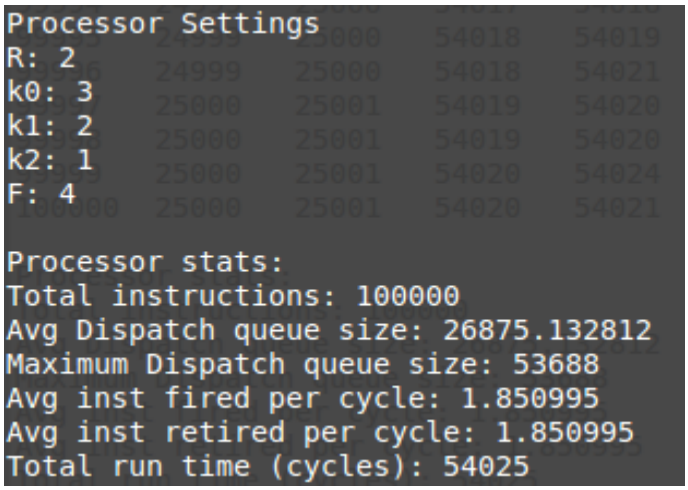
3] hmmer.100k.trace

As seen in the screenshot my outputs tally with the outputs provided for the default configuration:



4] mcf.100k.trace

As seen in the screenshot my outputs tally with the outputs provided for the default configuration:



Experimentation:

1] The number of Functional Units of each type were varied between 1 and 2 units of each type.

2] Fetch rate was varied between F=4 and F=8.

3] The number of result buses were varied in the following range: [1, (k0+k1+k2)]. The upper threshold was set as the total number of functional

units because even in the best case scenario, in a particular cycle all FUs would be ready to broadcast their results on the result buses, hence we would not need more than (k0+k1+k2) result buses.

1] gcc.100k.trace

The maximum value of IPC obtained was 2.422070. However, it is necessary to select the least amount of hardware that provides nearly (>95%) of the highest value for retired IPC. All the combinations having IPC greater than 95% (2.3009) have been shown in the table below:

Sr. No	Number of Result Buses (r)	Fetch Rate (f)	Number of k0 FUs	Number of k1 FUs	Number of k2 FUs	IPC
1	5	8	2	2	2	2.422070
2	4	8	2	2	2	2.411091
3	3	8	2	2	2	2.366752
4	5	4	2	2	2	2.422070
5	4	4	2	2	2	2.411091
6	3	4	2	2	2	2.366752

Inference:

- Amongst all the listed combinations the one highlighted in yellow seems to be most hardware efficient combination with IPC greater than 95% of the highest IPC. The ones with the highest IPC use more result buses (5) as compared with sixth one.
- Another point to be noted is that even though the third and sixth combinations have the same IPC we still opt for the sixth one because it has a lower fetch rate. Hence, we would require a smaller dispatch queue

and would also lead to fewer cache misses (a smaller cache could be used).

- I also analysed the impact of the number of functional units of each type and tabulated the results.

Number of k0 FUs	IPC
1	1.7279
2	2.3667

Table 1: Keeping k1=2, k2=2, f=4, r=3

Number of k1 FUs	IPC
1	2.0008
2	2.3667

Table 2: Keeping k0=2, k2=2, f=4, r=3

Number of k2 FUs	IPC
1	2.0488
2	2.3667

Table 3: Keeping k0=2, k1=2, f=4, r=3

From these results we can conclude that the number of instructions in this trace which use functional unit k0 are higher than those using k1 and k2 because when we keep k0 = 1 the IPC is at its lowest in the listed combinations. So in case of strict hardware restrictions (limited number of functional units) we could lower the values of k1 and k2 however it is advisable to keep k0=2.

2] gobmk.100k.trace

The maximum value of IPC obtained was 2.364457. However, it is necessary to select the least amount of hardware that provides nearly (>95%) of the highest value for retired IPC. All the combinations having IPC greater than 95% (2.2462) have been shown in the table below:

Sr. No	Number of Result Buses (r)	Fetch Rate (f)	Number of k0 FUs	Number of k1 FUs	Number of k2 FUs	IPC
1	6	8	2	2	2	2.364457

2	5	8	2	2	2	2.364457
3	4	8	2	2	2	2.361721
4	3	8	2	2	2	2.304625
5	5	4	2	2	2	2.364457
6	3	4	2	2	2	2.304625

Inference:

- Amongst all the listed combinations the one highlighted in yellow seems to be most hardware efficient combination with IPC greater than 95% of the highest IPC. The ones with the highest IPC use more result buses (5) as compared with sixth one.
- Another point to be noted is that even though the fourth and sixth combinations have the same IPC we still opt for the sixth one because it has a lower fetch rate. Hence, we would require a smaller dispatch queue and would also lead to fewer cache misses (a smaller cache could be used).
- I also analysed the impact of the number of functional units of each type and tabulated the results.

Number of k0 FUs	IPC
1	1.867100
2	2.304625

Table 1: Keeping k1=2, k2=2, f=4, r=3

Number of k1 FUs	IPC
1	2.027740
2	2.304625

Table 2: Keeping k0=2, k2=2, f=4, r=3

Number of k2 FUs	IPC
1	2.020774
2	2.304625

Table 3: Keeping k0=2, k1=2, f=4, r=3

From these results we can conclude that the number of instructions in this trace which use functional unit k0 are higher than those using k1 and k2 because when we keep k0 = 1 the IPC is at its lowest in the listed combinations. So in case of strict hardware restrictions (limited number of functional units) we could lower the values of k1 and k2 however it is advisable to keep k0=2.

3] hmmer.100k.trace

The maximum value of IPC obtained was 2.266906. However, it is necessary to select the least amount of hardware that provides nearly (>95%) of the highest value for retired IPC. All the combinations having IPC greater than 95% (2.1535) have been shown in the table below:

Sr. No	Number of Result Buses (r)	Fetch Rate (f)	Number of k0 FUs	Number of k1 FUs	Number of k2 FUs	IPC
1	5	8	2	2	2	2.266906
2	6	8	2	2	2	2.266854
3	3	8	2	2	2	2.206385
4	5	4	2	2	2	2.266906
5	4	4	2	2	2	2.262546
6	3	4	2	2	2	2.206385

Inference:

- Amongst all the listed combinations the one highlighted in yellow seems to be most hardware efficient combination with IPC greater than 95% of the highest IPC. The first and fourth combinations have the highest IPC of 2.266906 however combinations 3 and 6 lie within the 95% range of maximum IPC and also use lesser number of result buses.
- Now between the 3rd and 6th and combinations it would be pragmatic to opt for the one having a lower fetch rate so as to ensure a smaller dispatch queue and lesser number of cache misses while fetching instructions thereby leading to a smaller L1 cache.

Number of k0 FUs	IPC
1	1.692563
2	2.206385

Table 1: Keeping k1=2, k2=2, f=4, r=3

Number of k1 FUs	IPC
1	1.867379
2	2.206385

Table 2: Keeping k0=2, k2=2, f=4, r=3

Number of k2 FUs	IPC
1	1.924335
2	2.206385

Table 3: Keeping k0=2, k1=2, f=4, r=3

From these results we can conclude that the number of instructions in this trace which use functional unit k0 are higher than those using k1 and k2 because when we keep k0 = 1 the IPC is at its lowest in the listed combinations. So in case of strict hardware restrictions (limited number of functional units) we could lower the values of k1 and k2 however it is advisable to keep k0=2.

4] mcf.100k.trace

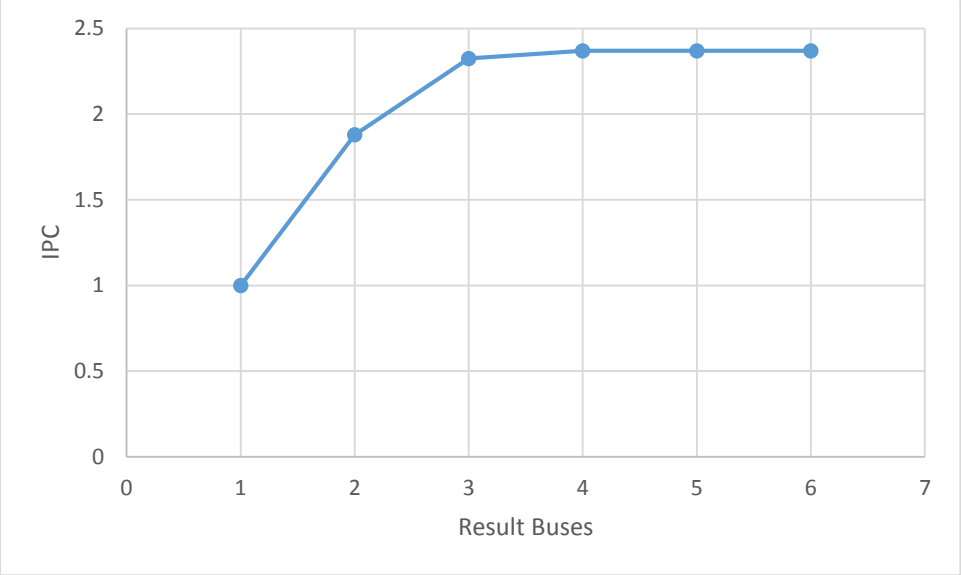
The maximum value of IPC obtained was 2.369444. However, it is necessary to select the least amount of hardware that provides nearly (>95%) of the highest

value for retired IPC. All the combinations having IPC greater than 95% (2.2509) have been shown in the table below:

Sr. No	Number of Result Buses (r)	Fetch Rate (f)	Number of k0 FUs	Number of k1 FUs	Number of k2 FUs	IPC
1	6	8	2	2	2	2.369444
2	5	8	2	2	2	2.369444
3	4	8	2	2	2	2.369444
4	3	8	2	2	2	2.324554
5	5	4	2	2	2	2.369444
6	4	4	2	2	2	2.369388
7	3	4	2	2	2	2.324500

Inference:

- Amongst all the listed combinations the one highlighted in yellow seems to be most hardware efficient combination with IPC greater than 95% of the highest IPC. The 1st, 2nd, 3rd and 5th combinations all have the highest IPC of 2.369444 however combinations 7 lies within the 95% range of maximum IPC and also use lesser number of result buses.
- We observe that for all combinations resulting in >95% IPC, the deciding factor becomes the number of result buses. To analyse this trend I plotted the variations in IPC against the change in the number of result buses.
- From Graph 1, it is evident that increasing the number of result buses beyond 3 does not provide a significant increase in IPC and the IPC still remains within 95% of its maximum value. So in order reduce hardware we limit the number of result buses to 3.



Graph 1: IPC v/s Number of Result Buses (F=4, k0=2, k1=2, k2=2)

CONCLUSIONS:

1] There was little or no difference in the IPC while changing the fetch rate from 4 to 8. Hence, it would be advisable to go for the lower rate so as have a smaller dispatch queue.

2] There was little or no difference in the IPC while changing the number of result buses from 3 to 6. Hence, keeping in mind hardware constraints it would be advisable to limit the number of result buses to 3.

4] On comparing the variation of IPC with the number of functional units it was observed that number of instructions using FU k0 as a functional unit were clearly greater than the others hence it would be advisable to always maintain a higher number of k0 functional units.