The following are the perf commands that I used to compare my implementation with the given implementation of division algorithm:

```
perf stat gcc -O3 -std=c99 p1.c -o p1_unoptimized
 perf stat gcc -O3 -std=c99 p1.c -o p1_optimized

 perf stat gcc -O3 -std=c99 p1_new.c -o p1_new_unoptimized
 perf stat gcc -O3 -std=c99 p1_new.c -o p1_new_optimized

perf stat -e L1-dcache-misses ./p1_unoptimized
perf stat -e L1-dcache-misses ./p1_optimized

perf stat -e L1-dcache-misses ./p1_new_unoptimized
perf stat -e L1-dcache-misses ./p1_new_optimized


perf stat -e L1-icache-misses ./p1_unoptimized
perf stat -e L1-icache-misses ./p1_optimized

perf stat -e L1-icache-misses ./p1_new_unoptimized
perf stat -e L1-icache-misses ./p1_new_optimized
```

1. The number of instructions decreased when compiler optimizations were used.
2. IPC increased going from the given implementation to the Integer division (unsigned) with remainder implementation.
3. Number of branches increases from the given implementation to the Integer division (unsigned) with remainder implementation.
4. D-cache misses decreases from the given implementation to the Integer division (unsigned) with remainder implementation.
5. I-cache misses increases from the given implementation to the Integer division (unsigned) with remainder implementation.