1 Pintools

The first part of this project was to use Pintools to analyze the TLB heiarchy of our processor. My processor only had one set of TLBs, one for instructions and another for data. This made it very hard to analyze how evective the TLB actually was. The TLB effectively reduces the number of memory access when using virtual addresses. The TLB stores virual address to physical address traslation in a small buffer and is used as a cache of sorts for this lookup. In theory, without a TLB every memory access could traslate to multiple memory accesses for the OS in order to traslate the address. Here in Table 1, you can see that dhrystone had to access the memory over 250 million times when accessing instructions. If there was no TLB, there would be upwards of 500 million memory access, and considering the latency for a modern processor to access RAM, this latency would build up and slow down the processor while its waiting for memory and not doing anything. In the results, you can see that there was only a small percent of a percent TLB miss, this means that the TLB algorithm is very efficient and removed the latency effect of virtual addresses. A similar result is obtained for the data TLB. The test was run 3 times and an average was taken.

	Test1	Test2	Test3	Average
Dps	38461.5	40000	38461.5	38974.33333
ITLB Hits	263245913	263245896	263245731	263245846.7
ITLB Miss	135	135	135	135
ITLB Total	263246048	263246031	263245866	263245981.7
DTLB Hits	124077146	124077142	124077084	124077124
DTLB Miss	269	255	269	264.3333333
DTLB Total	124077415	124077397	124077353	124077388.3