CPE 315 – Lab 4 Write Up

Alex Bozarth and Nicole Martin
March 13, 2014
Winter 2014

1. For shang compiled with no optimization, the number of cycles was 1,385,439,959 and the number of dynamic instructions was 1,165,086,982, thereby making the cycles per instruction (CPI) 1.189.

   For shang compiled with -O3 optimization, the number of cycles was 330,527,906 and the number of dynamic instructions was 318,956,344, thereby making the cycles per instruction (CPI) 1.036.

2. To predict branches, it was assumed that the next instruction in the instruction memory would be followed. Different decisions were not made for forward or backwards branches. If the compiler inserted a NOP immediately after a branch, it was not placing a useful instruction in the branch delay slot.

3. With no optimization, the GCC MIPS compiler did not use the branch delay slot at all, and overall had 0 useful instructions in the branch delay slot and 60,024,298 not useful instructions in the branch delay slot. In the case of the -O3 optimization, the GCC MIPS compiler had 50,146,455 useful instructions in the branch delay slot and 6,073,340 not useful instructions in the blank delay slot.

   When the GCC MIPS compiler actually optimizes the code, it makes excellent use of the branch delay slot since approximately 89% of the instructions placed in the branch delay slot were useful compared with the 0% of useful instructions in the case of no optimization.

4. The GCC MIPS compiler is excellent at avoiding load use hazards since, by definition, a true load use hazard wouldn't work with the way our MIPS pipeline is designed. The compiler automatically handles these by either inserting a NOP (a stall) or rearranging the instructions. With no optimization, there were 151,276,387 load use stalls. This is incredibly high compared with the 5,498,222 load use stalls in the -O3 optimized version, which is ......

5. You should chose the block size that has the best hit rate with high optimization. For shang, this would be a cache with a blocksize of 32 bytes. For -O3 optimization the 32 byte cache had the best hit rate of 83.7145%. It was not the best hit rate for the no optimization case, but it still had a 89.8965% hit rate, which is completely acceptable.

6. Compiling with no optimization makes the compiler very lazy. It does not rearrange instructions, inserts stalls after load hazards instead of rearranging instructions, and does not make any use of the branch or jump delay slots.