Phase 1 Report

Kerim Demir

28853

**Large tables method:**

In this case, when number of blocks is fixed, as block size is increased the number of misses decrease. This is because, when the array is accessed fort he first time it will be a compulsary miss; however because of the locality, we will be able to store the array in our blocks if the block size is equal to or bigger than the array size. The difference can be seen by fixing the number of blocks to a number and changing the block size from 4 to 128.

* Direct mapping/4-way set, 16 blocks with 128 word blocks, hit rate 699/700 ~ %100

**Small tables method:**

Same results can be observed by applying the same cache setup above. This can be because 4 different arrays are stored consecutively in the memory.

* Direct mapping/4-way set, 16 blocks with 128 word blocks 🡪 hit rate 699/700 ~ %100

**Non-linear function:**

I seperated the 16-bit number into 4-bit numbers with andi and sll-srl instructions. Then, each x\_i is treated as an index, so i added them to the base address of associated s-boxes then accumlated the result in $v0.

**Linear function:**

Because the shifts are circular, i also need the bits that are discarded when shifting, so i shift twice. For X <<< 6, i shift left 6 bits and shift right 10 bits and store the result in different registers, then i or those registers to get the circular shift result. X >>> 6 case is the same. Then i xor them in order and return.

**Permutation function:**

At each bit, i extract that bit into a different register with usign a register as mask. Then, i check if the bits corresponding mapping position is in the lef tor right side of it with a sle instruction. Accoring to the result, i subtract the index of the mapping from current index to get the shift amount and shift. This operation is in a loop, so it is repeated for each bit individually, then return.