Brian McIlwain

EE3752 Lab 3 Section 4

Purpose

The purpose of the experiment was to provide practice isolating manipulating specific bits in registers.

Problems

1) Add the high 16 bits from r1 and the low 16 bits from r2, place the word-size sum in r5; and (2) add the middle 16 bits from r3 and the middle 16 bits from r4, placing the word-size sum in r6.

2) r5 = sum from r1 and r2 as in part 1

r6 = sum from r3 and r4 as in part 1 // this is part 1

if (r0 == r5) // you will have 5 inputs in part 2 instead of 4

end program

else

r5 = sum of middle 16 bits from r1 and the middle 16 bits from r2

r6 = sum of high 16 bits from r3 and low 16 bits from r4

end program

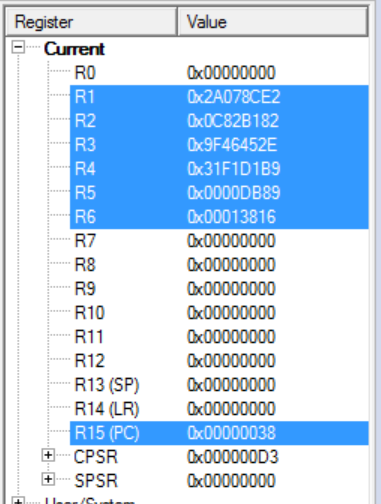
Method

1) I solved part 1 at first by using only shift left and shift right operations. I would move whatever subset of bits that I needed to add all the way to the left, then shift them right and use that to add to the running total. This solves problem 1, but when I got to part 2 I noticed that I needed the old value of the registers from part 1. My second approach was to use rotations and shift right. I’d rotate the bit string so that the bits that I was going to add were all the way to the left. Then I’d add the bits with an in-line shift right. After I had the total, I’d rotate the bit strings manipulated back to the way they were. This allowed me to preserve the original bit strings given.

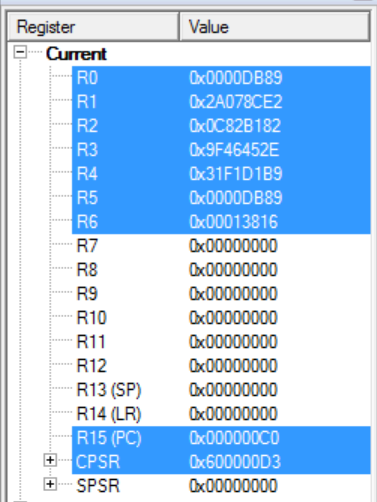
2) Part 2 was easy once I made the modification mentioned in part 1. The main problem was that I needed the original values in r1-4 so I modified my approach to accommodate this.

Results

1) With the given input, r5=0x0000DB89, and r6=0x00013816



2) When r0=0x0000DB89, r5=0x0x0000DB89, and r6=0x00013816 (exactly the same as part 1)



2) When r0=0x00000000, r5=0x00008A3D, r6=0x000170FF. Since r0 != r5, the extra lines of code are run, changing the values of r5 and 6.

