CSE3666 — Homework 2

Mike Medved

February 11th, 2022

1 Prompt

Loop unrolling is an optimization technique to improve the performance of programs. In the second implementation, we unroll the loop and process four array elements in A in each iteration. The unrolled loop in C is shown below. Translate the loop to RISC-V instructions. Try to minimize the number of instructions that are executed. Explain your code. How many instructions will be executed for the new loop?

```
for (i = 0; i < 100; i += 4) {
    B[i] = A[i] + 4;
    B[i+1] = A[i+1] + 4;
    B[i+2] = A[i+2] + 4;
    B[i+3] = A[i+3] + 4;
}
```

2 Deliverables

```
    # CSE 3666 Homework 2 - Question 1b
    .globl main
    .text

    main:
        li s1, 0           # i = 0
        li s4, 100         # i < 100
        beq x0, x0, test   # we know s1 < s4

    loop:
        slli t0, s1, 2 # t0 = i * 4

        # B[i] = A[i] + 4
        add t2, t0, s2 # compute addr of A[i]
        lw t1, 0(t2)   # retrieve and store A[i] in t1 register
        addi t1, t1, 4 # A[i] + 4
        add t3, t0, s3 # compute addr of B[i]
        sw t1, 0(t3)   # store value of t1 (A[i] + 4) into B[i]

        # B[i+1] = A[i+1] + 4
        # since we already have the address of A[i],
        # load the address+4 which corresponds to A[i+1]
        # since ints are 4-bytes.
        lw t1, 4(t2)
        addi t1, t1, 4 # A[i+1] + 4
        sw t1, 4(t3)   # store value of t1 (A[i+1] + 4) into B[i+1]
    # B[i+2] = A[i+2] + 4
    # knowing the address of A[i], add an offset of 8
    # to get the second element of the word array.
    lw t1, 8(t2)
    addi t1, t1, 4 # A[i+2] + 4
    sw t1, 8(t3)   # store value of t1 (A[i+2] + 4) into B[i+2]

    # B[i+3] = A[i+3] + 4
    # again, knowing the base address, add an offset of 12
    # to get the third element of the word array.
    lw t1, 12(t2)
    addi t1, t1, 4 # A[i+3] + 4
    sw 51, 12(t3)  # store value of t1 (A[i+3] + 4) into B[i+3]

    addi s1, s1, 4 # i += 4

test:
    bne s1, s4, loop
```
