ECE1110 Homework #1

Rachel Solak

September 20, 2021

1. Complete this procedure using RISC-V assembler.

<https://github.com/lukespowerconverters/ece1110/blob/main/Homework%20%231/hw1-assignment.asm>

2. How will you test this?

Convolution can be applied both ways, as (f\*g)(t) == (g\*f)(t), so we can test this formula by switching the order of the convolution. If it only works one way, then we know the code is broken. Also, we know that the graph of a continuous time convolution of two rectangular pulses is a triangle. We can test our code by inputting two rectangular pulses and graphing the outputs, ensuring that the convolution is a triangle.

3. What is your total cost in the inner loop?

|  |  |  |  |
| --- | --- | --- | --- |
| Instruction | Cost | Instruction | Cost |
| bgt t1, a0, donem | 2 | add s3, s3, s6 | 1 |
| mv t3, t1 | 1 | flw ft0, 0(s3) | 3 |
| addi a1, a2, -1 | 2 | sub t3, t0, t3 | 1 |
| mv a0, t0 | 1 | slli s6, t3, 2 | 2 |
| jal min | 1 | add s4, s4, s6 | 1 |
| la s3, h | 2 | flw ft1, 0(s4) | 3 |
| la s4, x | 2 | fmul.s ft1, ft0, ft1 | 1 |
| la s5, y | 2 | slli s6, t0, 2 | 2 |
| jal functiony | 1 | add s5, s5, s6 | 3 |
| addi t1, t1, 1 | 3 | flw ft0, 0(s5) | 3 |
| j testm | 1 | fadd.s ft0, ft0, ft1 | 1 |
| slli s6, t3, 2 | 2 | fsw ft0, 0(s5) | 3 |
|  |  | ret | 1 |

The total cost of the inner loop is 45.

4. DSP microprocessors include a *multiply-accumulate* instruction. What would such an instruction look like in RISC-V? How will this change the total cost in the inner loop?

In RISC-V, the multiply-accumulate instruction is as follows:

fmadd.s ft0, ft1, ft2, ft3 #ft0 = ft1 \* ft2 + ft3

The product of the middle 2 registers (ft1 + ft2) multiplied by the last register (ft3) is stored in the first register (ft0); a .s is added to the end of the instruction fmadd for single precision, while .d is added for double-precision.

This would decrease the total cost of the inner loop, as the product of h[m] and x[n-m] and the sum of y[n] and that product could be obtained in the same line. Instead of:

fmul.s ft1, ft0, ft1 #ft1 = h[m] \* x[n-m]

fadd.s ft0, ft0, ft1 #ft0 = y[n] + (h[m] \* x[n-m])

These two lines could be combined into one:

fmadd.s ft0, ft1, ft2, ft3 #ft0 = y[n] = h[m] \* x[n-m] + y[n]

The cost of two original lines was 2, and the cost of new and improved line only 1.