Samuel Kan

Professor Jungwirth

CMSC 411

10/19/2021

Project

Function for hpf below:

int hpf(int in)

{

    static int w\_old = 0;

    int w = in + 0.5 \* w\_old;

    int out = w - w\_old;

    w\_old = w;

    return out;

}

Excel results screenshot below:

I will include the C file and the excel file in the email. The MIPS code for the function is shown below:

HPF: SRA $12, $10, 1 #R12 = w\_old / 2

ADD $13, $6, $12 #R13 = w, w = R6 (input) + R12 (old\_w / 2)

SUB $9, $13, $10 #R9 = output, output = w - w\_old

# w\_old = w;

ADD $10,$13,$0 #

Here is a screenshot of the 7th, and 10th output for the loop, which is stored in register R9:

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generatedThe correct output for an input array of {-5, 3, 2, 7, -9, 20, 10, 3, 2, 1} is -5, 5, 2, 6, -13, 22, 1, -6, -4, -3.

I didn’t want to screenshot all the values as that would make this document excessively long. The code for the entire thing is at the bottom of the document. Excel plot for input is below:

Based on the peaks, I would hazard a guess at 3 kernels of popcorn as there are three positive peaks?

# Tell assembler to not insert instructions to fill branch delay slots.

# This is necessary when branch delay slots are disabled.

.set noreorder

.global \_start

\_start:

START: ADDI $1,$0,0x100 # base address for input array[ ]

ADDI $2,$0,0x140 # base address for output array [ ]

ADDI $3,$0,0 # loop index initialized = 0

ADDI $4,$0,40 # loop last index value = 40; // count by 4 bytes

# Let R10 = w\_old value. Initalize to zero

ADDI $10,$0,0 # set w\_old = 0;

LOOP: BEQ $3,$4,DONE # is R3 = R4 = last index

ADD $5,$1,$3 # R5 = input array[ ] base address + index

LW $6,0($5) # R6 = load word from R5

# place your MIPS filter code here

# let R9 contain filter ouput

#WE CAN USE REGISTERS STARTING FROM 11

SRA $12, $10, 1 #R12 = w\_old / 2

ADD $13, $6, $12 #R13 = w, w = R6 (input) + R12 (old\_w / 2)

SUB $9, $13, $10 #R9 = output, output = w - w\_old

# w\_old = w;

ADD $10,$13,$0 #

# place result in R6 and then save in output array []

ADD $7,$2,$3 # R7 = putput array[ ] base address + index

SW $6,0($7) # Mem(R7) = R6

ADDI $3,$3,4 # advance index by 4 bytes

J LOOP # jump to loop

DONE: NOP