**Week 5**

**NAME CHETAN P MURTHY**

**SRN PES2UG22EC042**

1. **Write an assembly program to check whether a given number in an array of elements is divisible by 9**

.data

n: .word 0x000000009, 0x00000135 # Given number to check divisibility by 9

.text

la x5, n # Load address of the number into x5

lw x6, 0(x5) # Load the value of n into x6

addi x7, x0, 0x0F # Mask for the lower 4 bits (0x0F = 00001111)

addi x28, x0, 0 # Initialize sum of digits to 0

addi x29, x0, 8 # We have 8 digits to process (since 32-bit number, 8 digits of 4-bits)

loop:

and x18, x7, x6 # Isolate the lower 4 bits of x6 (digit)

add x28, x28, x18 # Add the isolated digit to the sum

srli x6, x6, 4 # Right shift x6 by 4 to process the next digit

addi x29, x29, -1 # Decrement loop counter

bne x29, x0, loop # Repeat the loop if there are more digits to process

new:

# Now we have the sum of the digits in x28

# Check divisibility by subtracting 9 repeatedly

addi x30, x0, 9 # Load 9 into x30 for subtraction loop

check\_divisibility:

blt x28, x30, done # If sum is less than 9, exit loop

sub x28, x28, x30 # Subtract 9 from the sum

j check\_divisibility # Repeat until sum is less than 9

done:

# If the final sum is 0, it's divisible by 9

beq x28, x0, divisible

not\_divisible:

li x31, 0 # Set x31 to 0 (not divisible by 9)

j end

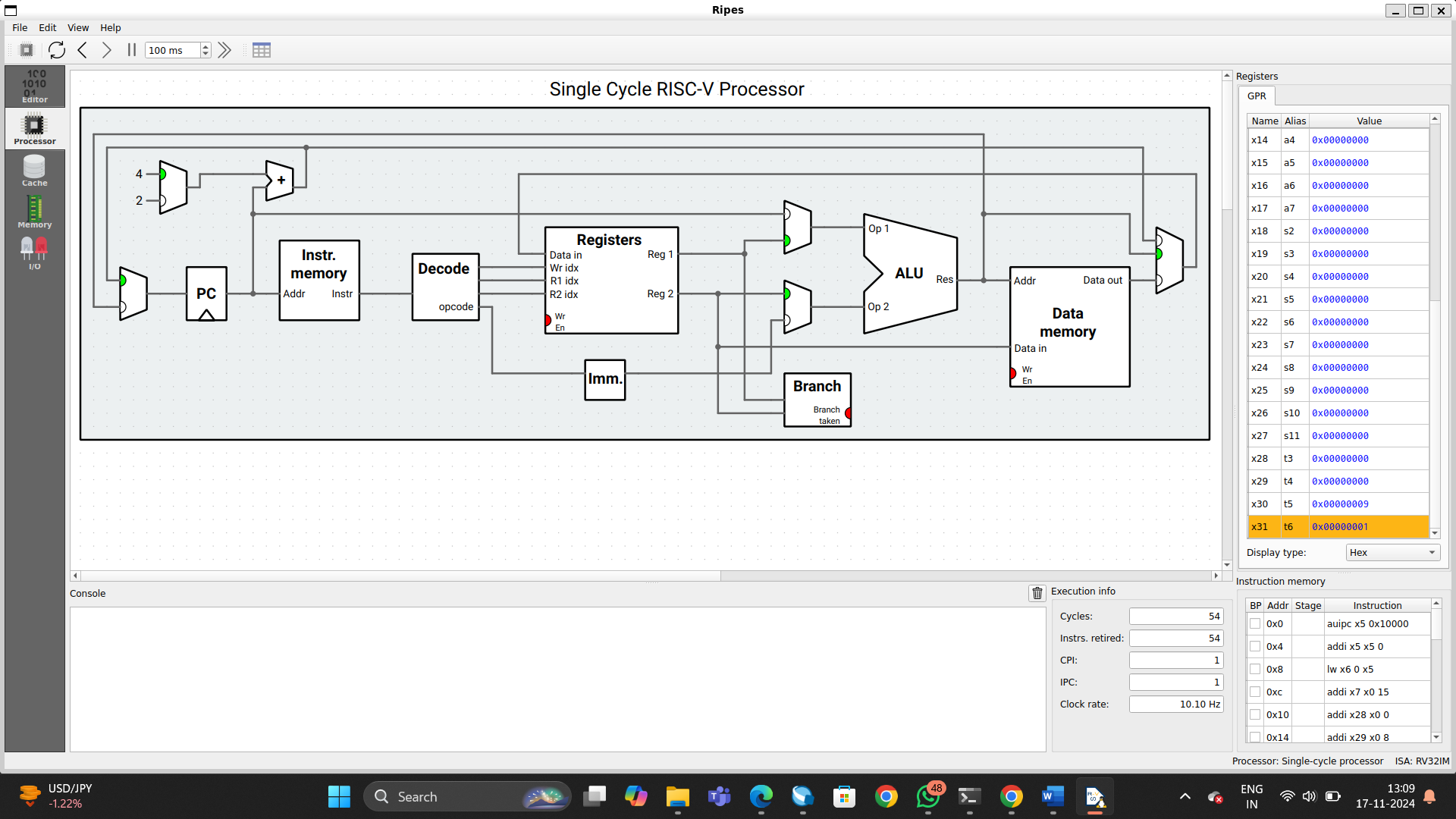
divisible:

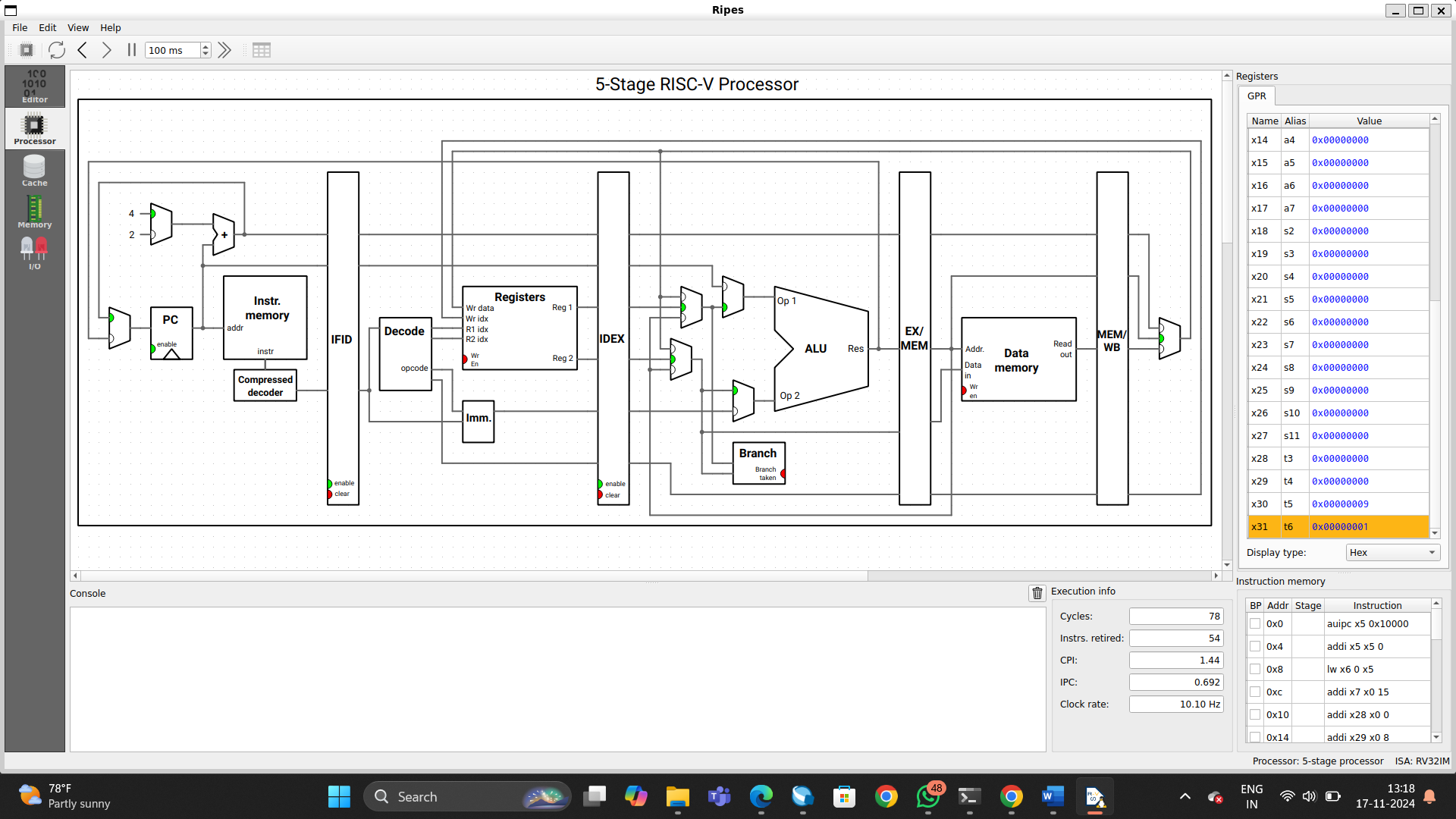
li x31, 1 # Set x31 to 1 (divisible by 9)

end:

# x31 contains the result: 1 if divisible by 9, 0 otherwise

nop





1. **Write an Assembly Program for the following C code:**

main() { unsigned short int a[11] = {0x1234, 0x5678, ...}; unsigned short int b[11] = {0x1234, 0x5678, ...}; unsigned short int c[11] = {0x1234, 0x5678, ...}; for(i = 0; i < 10; i++) { c[i] = a[i] \* b[i] + c[i-1]; } } ```

.data

a: .half 0x1234, 0x5678, 0x1111, 0x2222, 0x3333, 0x4444, 0x5555, 0x6666, 0x7777, 0x8888, 0x9999 # Array a

b: .half 0x1234, 0x5678, 0x1111, 0x2222, 0x3333, 0x4444, 0x5555, 0x6666, 0x7777, 0x8888, 0x9999 # Array b

c: .half 0x1234, 0x5678, 0x1111, 0x2222, 0x3333, 0x4444, 0x5555, 0x6666, 0x7777, 0x8888, 0x9999 # Array c

.text

main:

la x10, a # Load base address of array a into x10

la x11, b # Load base address of array b into x11

la x12, c # Load base address of array c into x12

li x14, 10 # Loop counter (i = 0 to 9)

li x13, 0 # i = 0, loop index

loop:

beq x13, x14, exit # If i == 10, exit the loop

lh x15, 0(x10) # Load a[i] into x15

lh x16, 0(x11) # Load b[i] into x16

mul x17, x15, x16 # Multiply a[i] and b[i], result in x17

beq x13, x0, first\_iter # If i == 0, skip adding c[i-1]

lh x18, -2(x12) # Load c[i-1] into x18

add x17, x17, x18 # Add c[i-1] to the result

first\_iter:

sh x17, 0(x12) # Store result into c[i]

addi x10, x10, 2 # Move to a[i+1]

addi x11, x11, 2 # Move to b[i+1]

addi x12, x12, 2 # Move to c[i+1]

addi x13, x13, 1 # Increment loop index

j loop # Jump back to loop

exit:

nop # Exit point

