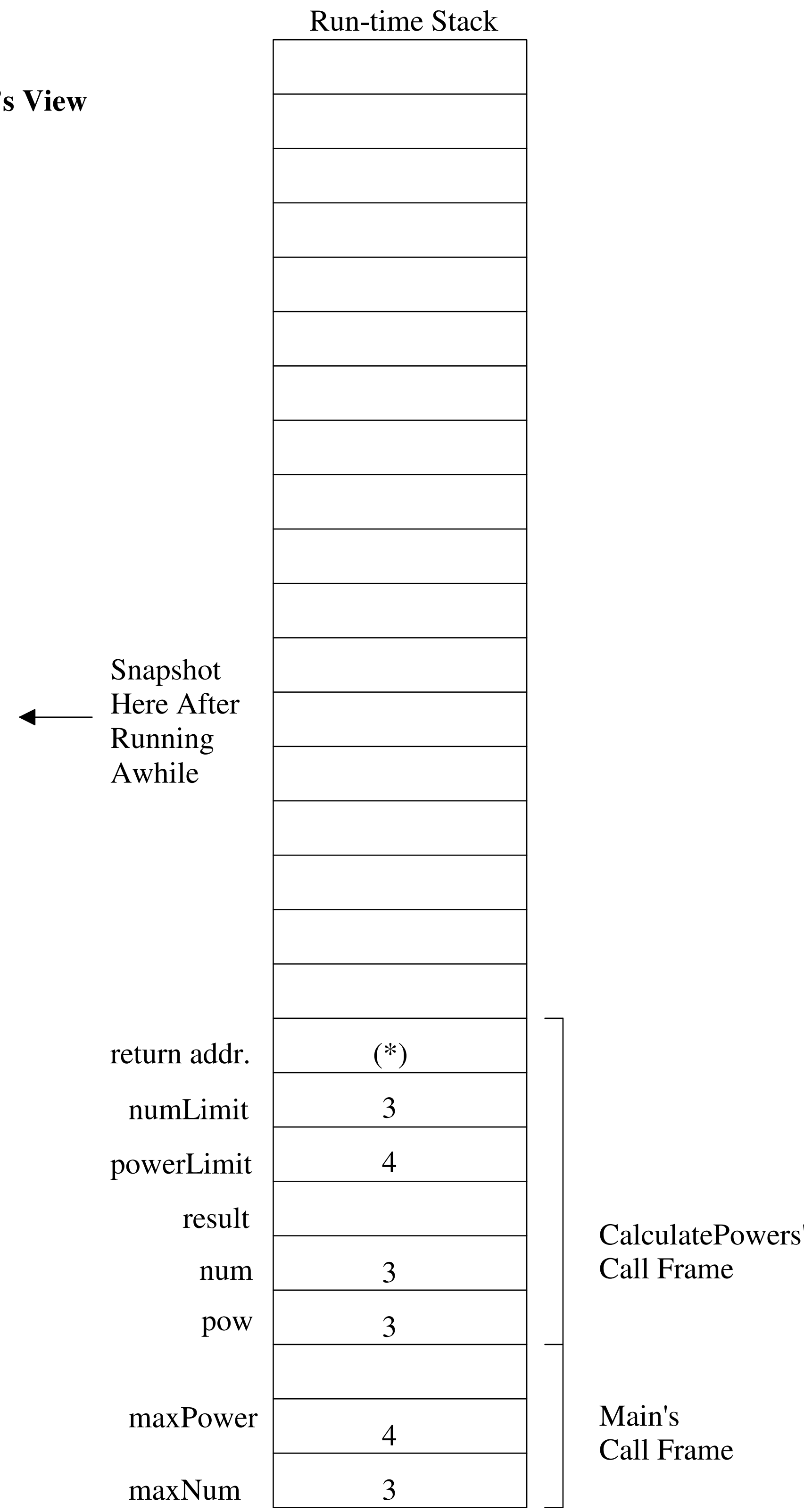
Name:__________________________

High-level Language Programmer's View

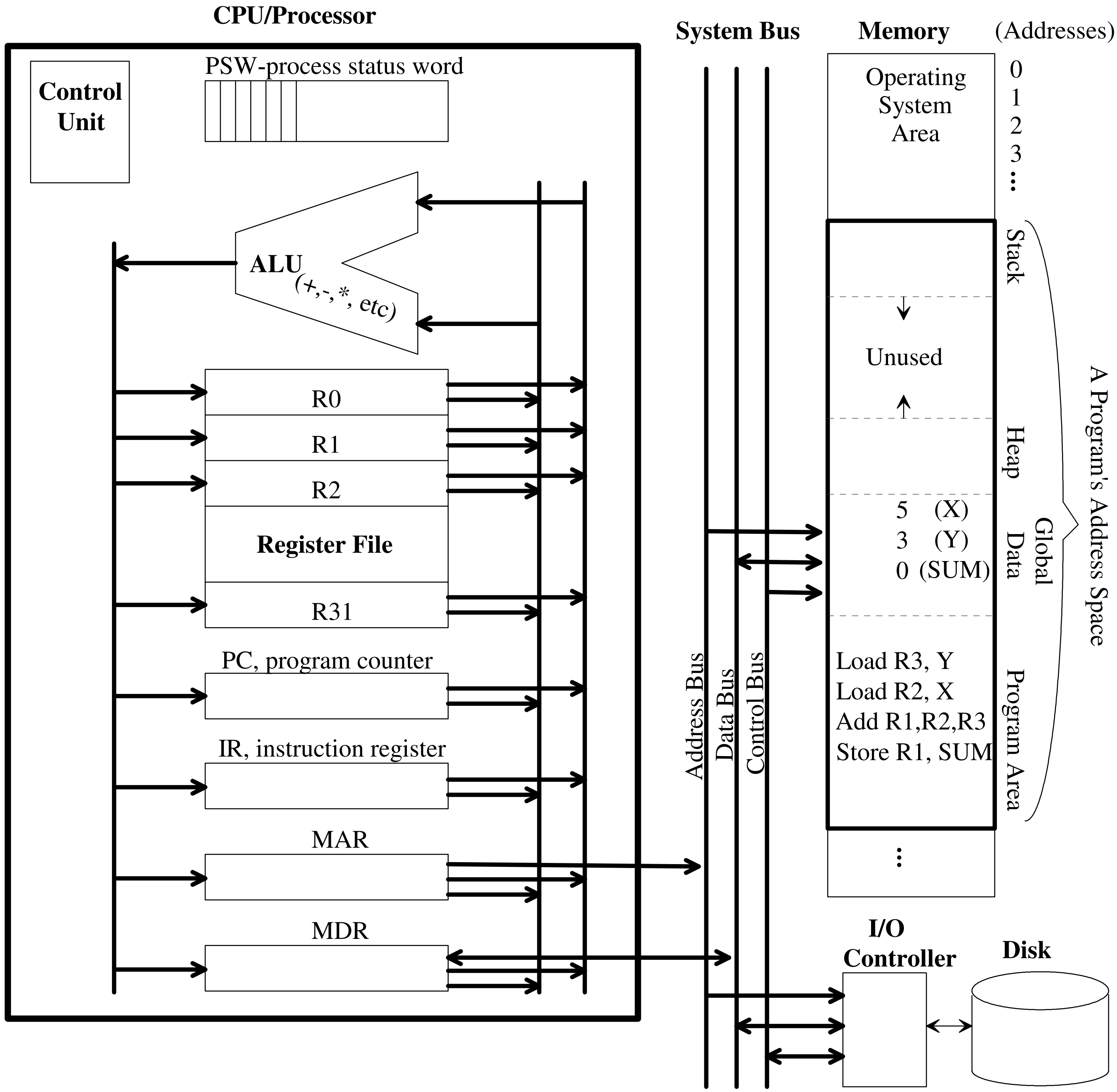
```
main:
maxNum = 3
maxPower = 4

CalculatePowers(maxNum, maxPower)
(*)
. . .
end main


CalculatePowers(In: integer numLimit,
                    integer powerLimit)
integer num, pow, result

for num := 1 to numLimit do
     for pow := 1 to powerLimit do
         Power(num, pow, result)
(**)     print num " raised to " pow " power is "
                  result
     end for pow
end for num

end CalculatePowers


Power( In:  integer n, integer e, Out: result)

if e = 0 then
     result = 1
else if e = 1 then
     result = n
else
     Power(n, e - 1, result)
     result = result * n          (***)
end if

end Power
```



1) Trace the execution of the recursive function Power by showing the run-time stack.

2) What is the most number of call frames on the stack for the **whole** program?

Name:________________________



Instruction/Machine Cycle of stored-program computer - repeat all day

1. Fetch Instruction - read instruction pointed at by the program counter (PC) from memory into Instr. Reg. (IR)
2. Decode Instruction - figure out what kind of instruction was read
3. Fetch Operands - get operand values from the memory or registers
4. Execute Instruction - do some operation with the operands to get some result
5. Write Result - put the result into a register or in a memory location

3) What has to happen to the PC during the instruction cycle?

Name:__________________________

Assembly-language Programmer's View

4) Trace the hypothetical RISC-like assembly language program and indicate the resulting value of the registers R1, R2, R3, and R4.

```
        .data                                   # COMMENTS - Initial Global Data values
X:      .WORD       2                           # variable X initialized at assembly time to 2
Y:      .WORD       3                           # variable Y initialized at assembly time to 3
Z:      .WORD       0                           # variable Z initialized at assembly time to 0

        .program
Begin:
        LOAD R1, X                              # loads X's value into register R1
        LOAD R2, Y
        CLEAR  R3                               # sets R3's value to 0
        MOVE R4, R2                             # R4 := R2
Loop:
        ADD   R3, R3, R1                        # R3 := R3 + R1
        SUB_IMMEDIATE  R4, R4, #1               # R4 := R4 - 1
        BRANCH_GREATER_THAN_ZERO R4, Loop       # if R4 > 0 then goto Loop label
        STORE  R3, Z                            # store R3's value into variable Z
End:
```

	R1	R2	R3	R4
Resulting register values				

a) What is the resulting value in Z?

b) What calculation does this code perform?

5) During the execution of the above assembly language code: (Assuming no cache)

a) How many memory reads were performed? (state any assumptions)

data reads =

instruction reads (assume one read per instruction-fetch) =

b) How many memory writes were performed? (state any assumptions)

6) Why would somebody write assembly language code?

7) How fast is each of the following levels of memory?

a) register:

b) level 1 (L1) of cache:

c) main memory (RAM):

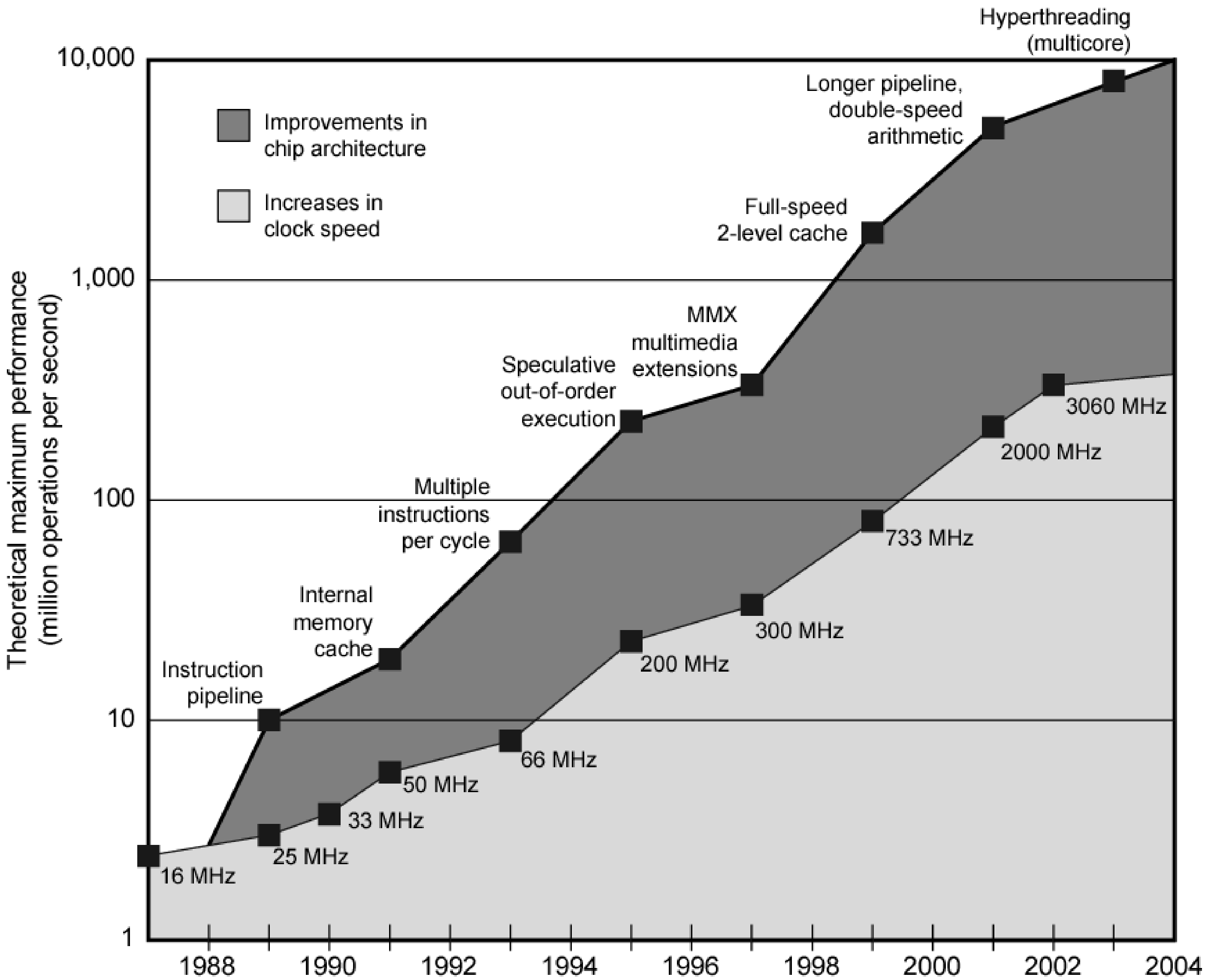
d) hard drive:

Name:_________________________

8) "Moore's law" (Gordon Moore - cofounder of Intel) - predicts that the number of transistors that could be put on a single chip would double every year (later changed to 18 months).

a) What kind of curve (# transistors vs. time) does Moore's law predict? (linear, quadratic, exponential, etc.)

b) As gate density increases on a chip, why would clock speed increase?



9) From the above graph, what architectural improvements seem to have the biggest impact on performance?

10) What architectural improvements have you observed "recently" to increase processor performance?