1. **Timer and I/O Interrupts** (10 pts)
Consider a multicore or multiprocessor computer with 4 cores (processors) that share memory, I/O devices, and OS code. Assume processes are eligible to run on any of the four cores at all times.

a. For the timer interrupt system, how many timer clocks are needed for best performance? Circle your answer and explain in one sentence or less.

   One clock              Four clocks              Doesn't matter how many clocks

b. For the I/O interrupt system, on which CPU(s) should the interrupt occur? Circle your answer and explain in one sentence or less.

   On one specific CPU              On any single CPU                 On all four CPUs

2. **Processes and Threads** (10 pts)
For each statement below, circle P if the statement is true for processes and circle T if it is true for threads. Some statements may be true for both; some may not be true for either.

a. PC keeps track of next instruction to be executed
   P    T
b. Has high context switch overhead
   P    T
c. Can be CPU-bound or I/O bound
   P    T
d. Can only be scheduled by the OS kernel
   P    T
e. Its ID is only known by the kernel
   P    T
f. Can fork a child
   P    T
g. Has a unique CPU state associated with it
   P    T
h. Useful for embarrassingly parallel programs
   P    T
3. **fork and exec** (5 pts)
What code is running after this code is executed? What can you eliminate and still have the same effect? Circle the answer and explain.

```c
main() {
    pid_t pid1, pid2;
    pid1 = exec("/bin/mycode");
    pid2 = fork();
    if (pid2 == 0) exec("/bin/childcode");
    else exec("/bin/parentcode");
}
```

/bin/mycode /bin/childcode /bin/parentcode none