CS 4/53201 # Project #1 Solution **Operating Systems**

1. (40 points) These questions are concerned with the Nachos operating system.

a. When you compile as instructed, and run Nachos, why does the "–z" argument do something, but the "–x" argument doesn't work?

In main.cc, the "-z" argument (which prints the copyright notice) is always compiled. In contrast, the "-x" argument (which should run a users program) is only compiled if nachos is compiled using the "USER_PROGRAM" switch.

As we can see below, when nachos is compiled as instructed, the "THREADS" switch is used (which is why ThreadTest() runs), but the "USER_PROGRAM" switch is not used.

cd threads; /local/opt/gcc/bin/make depend g++ -I../threads -I../machine -DTHREADS -DHOST_SNAKE -DHOST_IS_BIG_ENDIAN -DCHANGED ...

b. Where and how is the CPU scheduler started?

The CPU scheduler is started as "scheduler", an instance of class Scheduler, in the function Initialize in system.cc.

c. What does ThreadTest do?

It generizes a new thread "t" with a debug name of "forked thread" which is to call SimpleThread() with an argument of 1. Then ThreadTest() continues by calling SimpleThread() itself with an argument of 0. The result of this action a "ping-pong" switching between the two threads, the current main thread and the newly forked thread.

d. What is in a thread's "thread control block"?

A thread's "thread control block" is specified in class Thread in "thread.h". A thread control block includes the thread name, status (running/ready/blocked), the stack and a pointer to the bottom of the stack, stackTop (the current stack pointer), space to save CPU registers while the thread is not running, the user code of the thread (if needed).

2. (40 points) These questions are concerned with the emulated machine that runs underneath the Nachos operating system.

a. What CPU state is saved during a context switch? What function does this, and how many registers are saved?

CPU register state is saved during a context switch.

Two possible answers, full credit for either one, extra credit for anyone who gives both:

(1) The function Thread::SaveUserState in thread.cc saves the user registers, saving NumTotalRegisters, which is defined to be 40 in machine/machine.h.

(2) The SWITCH function in the thread/switch.s does this. SPARC and MIPS processors have 10 registers each, but the Snake has 18. For example, 10 register states are saved. They are: 8 callee registers (S0 through S7), Frame Pointer(FP) and Program Counter(PC).

b. What does a hardware timer do? How is the Nachos hardware timer different from a "real" timer?

A hardware timer generates a CPU interrupt every X milliseconds. Unlike real hardware, the Nachos timer does NOT advance its simulated time anywhere in the code where interrupts are enabled, but only when a user instruction is executed, or when interrupts are re-enabled.

c. What function allocates space for the main memory of the machine?

In the file machine/machine.cc, the function Machine::Machine() allocates space for the main memory of the machine. (mainMemory= new char[MemorySize];)

d. What does the MIPS ADDU instruction do?

It adds two numbers and saves the result into another register without overflow detection.

3. (20 points) Compile and run Nachos and observe the output. Then modify "ThreadTest" to fork a second thread "u" named "second forked thread" immediately after it forks thread "t" named "forked thread". What are the results when Nachos is compiled and run? Why does thread 0 run first, instead of thread 1, give that the fork of thread 1 occurs in ThreadTest before it calls SimpleThread(0)?

Output not shown here…

Although thread 1 is forked at an earlier time, the scheduler just puts it at the end of the ready-queue because the current thread is running. Thread 1 has to wait until thread 0 (the main thread) yields the CPU in function SimpleThread().