Rutwij Patel
Imperial College London

Pintos Task 0

1. git clone git@gitlab.doc.ic.ac.uk:lab2324_autumn/pintos_task0_ or can use git clone git@gitlab.doc.ic.ac.uk:lab2324_autumn/pintos_task0_rp1222.git for ease of use (since you don't have to type in username and password for future clone, push or pull operations).

2. Using strcpy() can overflow the buffer reserved for its output. This can be dangerous when overwriting memory beyond the bounds of the array (buffer) since this can cause undefined behaviour. Instead, the safer alternative is to use strlcpy();

3.
    a. The result file is: alarm-multiple.result
    b. The result log can be found at: src/devices/build/tests/devices/alarm-multiple.result
    c. The output file is: alarm-multiple.output
    d. The output log can be found at: src/devices/build/tests/devices/alarm-multiple.output

4.
    a. The size of the struct must be less than 1kB otherwise there will not be sufficient space for the kernel stack to grown downwards. The total size of the execution stack must be less than 4kB.
    b. If the combined size of the struct and execution stack are larger than 4kB a stack overflow occurs which will corrupt the thread state. To identify if a stack overflow has occurred, the thread_current() function compares the value of the 'magic' member of the running thread. The magic member is a constant arbitrary value defined in THREAD_MAGIC. When a stack overflow occurs, the contents of the thread are usually changed (which means the magic member is also changed). When compared, if these values are not equal, this means stack overflow has occurred.

5. A context switch refers to the process of switching from execution of one particular thread to another. The process is defined below:
    a. Main() initialises the thread system using the thread_init() function. This creates the first thread structure.
    b. The thread_start() function is called, which creates an idle thread. This also enables interrupts and subsequently the scheduler using intr_yield_on_return().
    c. The thread_create() function creates and starts a new thread by allocating a page for the thread's struct thread and stack before initialising its members. Then it sets up a set of fake stack frames. When initialised, the thread starts in the 'blocked' state and is 'unblocked' just before returning to allow a new thread to be scheduled in its place.
    d. At this point, 3 things can happen to the running thread.

   i. Thread_block() changes the state of the running thread to a blocked state until it is unblocked.
   ii. Thread_exit() causes the current thread to exit execution.
   iii. Thread_yield() yields the CPU to the scheduler which will allow a new thread to start execution.

e. At this point, the schedule() function is responsible for switching to the next thread. It records the current thread in a local variable and determines the next thread to run by calling next_thread_to_run() before switching to it via the switch_threads() function.
   i. Switch_threads() saves registers on the stack, saves the current CPU's stack pointer in the currently running threads struct and restores the new thread's stack into the CPU stack pointer.
   ii. Thread_schedule_tail() then activates the new thread

Interrupts are used to stop and alert the CPU of some activity that stops normal execution. If interrupts are on, this means the currently running thread may be pre-empted by another thread at any given time. Examples of interrupts include user input or computation errors.

6. The default length of a scheduler time slice is 100 ticks per second. There are 4 ticks per time slice, so each scheduler time slice is 0.04 seconds long.

7. #include <inttypes.h>

   uint64 value = …;

   printf("value = %" PRIu64 "\n", value);

The only necessary header file is <inttypes.h> since it provides a way to format the types within printf. The PRIu64 macro formats the uint64 value as required.

8. Reproducibility is the property of programs that states that the same result can be produced with the same input values. Running Pintos in QEMU is not deterministic (which means running the same code may return different results even with the same inputs). This means that Pintos is not reproducible. The reason for this is because time interrupts can happen at irregular intervals. To prevent missing bugs, run tests multiple times to increase confidence.

9. A semaphore is a non-negative integer that has 2 atomic operations:
   a. "Down" which waits for the value to be positive before decrementing it.
   b. "Up" which increments the value.

   For example, if a thread A is waiting for a thread B to complete some activity it can pass a semaphore intialised to 0 to B before using the "down" operation. Once B completes the activity, it will "Up" the semaphore signalling to A that the job is now complete.

Rutwij Patel
Imperial College London

A lock is similar to a semaphore which is initialised with a value of 1. The “Up” operation is called “Release” and the “Down” operation is called “Acquire.”

Locks have an extra property that only the “owner” of a thread (the one that acquires the thread) is allowed to release it.

10. A race condition occurs in multi-threaded programs when two (or more) threads attempt to access the same resource simultaneously. Atleast one of these threads will be trying to perform a ‘write’ operation on the data. The order in which threads execute may not necessarily be the same for each test, therefore this can lead to undefined behaviour.

    The test case if(x != null) is insufficient since the operation is non atomic, so x may be accessed and modified by another thread before complete execution of the condition. This may lead to unexpected answers.

    To ensure thread safety, a lock or a mutex can be used to guard access to the variable x.