This is a closed book/notes examination. Write the answers directly on this paper. You are not allowed to use any other paper. Make your answers as concise as possible. You have one hour and 25 minutes to finish your exam. Please, use the time wisely.

GOOD LUCK!

And remember: DON’T PANIC!
1. Process/thread management (total 20 points).

   a. (5 points) Draw a diagram of possible state transitions for a process (possible states are RUNNING, READY-TO-RUN, BLOCKED, and TERMINATED/ZOMBIE).

   b. (5 points) What are the steps an operating system has to perform during the context switch between two processes (if it uses paging for memory management)?

   c. (5 points) Is context-switching between two threads less expensive than switching between two processes? Why?

   d. (5 points) A brand new SGI Altix Intel Itanium system has 64 CPUs running Linux OS. What is the maximum number of processes in RUNNING state at a single time when it executes 200 processes? What is the maximum number of processes in BLOCKED state?
2. CPU scheduling (total 20 points).

a. (10 points) All processes arrive in numerical order at time 0 with the following running times: P1 300ms, P2 50ms, P3 200ms. If the time-slice quantum is 50ms, show the scheduling order for these processes under FIFO (First-In-First-Out), SJF (Shortest-Job-First), and RR (Round-Robin) CPU scheduling algorithm.

<table>
<thead>
<tr>
<th>Time [ms]</th>
<th>FIFO</th>
<th>SJF</th>
<th>RR</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>50</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>100</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>150</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>250</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>300</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>350</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>400</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>450</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>500</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

b. (5 points) Which scheduling policy in (a) has the best average turnaround time (difference between the process’ completion and arrival time).

c. (5 points) What is the difference between preemptible and non-preemptible scheduling? What would be the waiting time for a high-priority process which arrives at time 214ms if the system in (a) uses preemptible scheduler? What if the scheduler is non-preemptible? (for this problem assume that the context switch requires 0.01ms.)
3. Deadlocks (total 15 points)

An embedded video streaming system has three resources: video capture card (R1), video display card (R2), and network card (R3). In order to implement zero-copy streaming, designers implemented two concurrent processes. Process P1 needs to allocate all three resources to be able to simultaneously stream captured video over network and display it on the monitor. Process P2 needs to allocate network and video display card to be able to directly display data received over the network card.

a. (10 points) P1 allocates resources in the following order: R1, R2, R3. P2 allocates resources in the following order: R3, R2. Draw the resource-allocation graph that shows that the deadlock in this system is possible.

b. (5 points) Propose a method which will guarantee that deadlock cannot occur.
4. Synchronization (total 25 points)

a. (10 points) Designers decided to use value 1 to represent that a lock is not acquired and value 0 to represent that the lock is acquired by some thread. Implement constructor Lock(), Acquire(), and Release() methods for Lock class using the atomic SWAP operation. Write C++-like pseudo code.

b. (15 points) Synchronize producer and consumer threads that use 10-item buffer for communication. Write pseudo code for both threads. Assume that you can use void Buffer::store(Item item) and Item Buffer::get() methods, but you need to explicitly ensure that you never store more than 10 items in the buffer (to prevent overflow).
5. Memory management (total 20 points)

A 32-bit system uses two-level paging for address translation. Logical addresses are split into three parts: 4-bit first-level (outer) page number, 12-bit second-level page number, and 16-bit offset.

a. (5 points) What is the size of memory page for this system? How do you know?

b. (10 points) Depict the address translation by drawing relevant page tables for a process if you know that its logical address \texttt{0x00010023} resides at physical location \texttt{0xFFFF0023} and logical address \texttt{0xF0000333} resides at physical location \texttt{0x00010333}.
c. (5 points) If the system with two-level paging uses TLB, and has TLB hit rate of 0.9, calculate the effective memory access time for this system. Assume that TLB access requires 10\text{ns} and each memory access requires 100\text{ns}. 
Relax. This is just an article for those of you who finish early and have nothing else to do.

COMPUTERWORLD 1 August:

CREATORS ADMIT UNIX, C HOAX

In an announcement that has stunned the computer industry, Ken Thompson, Dennis Ritchie and Brian Kernighan admitted that the Unix operating system and C programming language created by them is an elaborate April Fools prank kept alive for over 20 years. Speaking at the recent UnixWorld Software Development Forum, Thompson revealed the following:

"In 1969, AT&T had just terminated their work with the GE/Honeywell/AT&T Multics project. Brian and I had just started working with an early release of Pascal from Professor Nichlaus Wirth's ETH labs in Switzerland and we were impressed with its elegant simplicity and power. Dennis had just finished reading 'Bored of the Rings', a hilarious National Lampoon parody of the great Tolkien 'Lord of the Rings' trilogy. As a lark, we decided to do parodies of the Multics environment and Pascal. Dennis and I were responsible for the operating environment. We looked at Multics and designed the new system to be as complex and cryptic as possible to maximize casual users' frustration levels, calling it Unix as a parody of Multics, as well as other more risque allusions. Then Dennis and Brian worked on a truly warped version of Pascal, called 'A'. When we found others were actually trying to create real programs with A, we quickly added additional cryptic features and evolved into B, BCPL and finally C. We stopped when we got a clean compile on the following syntax:

```
for(;P("\n"),R--;P("!"))for(e=C;e--;P("~"+(u++/8)%2))P("|"+(u/4)%2);
```

To think that modern programmers would try to use a language that allowed such a statement was beyond our comprehension! We actually thought of selling this to the Soviets to set their computer science progress back 20 or more years. Imagine our surprise when AT&T and other US corporations actually began trying to use Unix and C! It has taken them 20 years to develop enough expertise to generate even marginally useful applications using this 1960's technological parody, but we are impressed with the tenacity (if not common sense) of the general Unix and C programmer. In any event, Brian, Dennis and I have been working exclusively in Pascal on the Apple Macintosh for the past few years and feel really guilty about the chaos, confusion and truly bad programming that have resulted from our silly prank so long ago."

Major Unix and C vendors and customers, including AT&T, Microsoft, Hewlett-Packard, GTE, NCR, and DEC have refused comment at this time. Borland International, a leading vendor of Pascal and C tools, including the popular Turbo Pascal, Turbo C and Turbo C++, stated they had suspected this for a number of years and would continue to enhance their Pascal products and halt further efforts to develop C. An IBM spokesman broke into uncontrolled laughter and had to postpone a hastily convened news conference concerning the fate of the RS-6000, merely stating 'VM will be available Real Soon Now'. In a cryptic statement, Professor Wirth of the ETH institute and father of the Pascal, Modula 2 and Oberon structured languages, merely stated that P. T. Barnum was correct.

In a related late-breaking story, usually reliable sources are stating that a similar confession may be forthcoming from William Gates concerning the MS-DOS and Windows operating environments. And IBM spokesman have begun denying that the Virtual Machine (VM) product is an internal prank gone awry.

And, it was just a joke. Authors hoped it would be funny. I don’t know if you agree with them after debugging your pointer problems in project 2.