CS 6210 Spring 2013 Midterm Solution

Name:_______Kishore________________________GT Number:

Monday February 25, 20113 (11:05 to 11:55 AM)

Note:

1. **Write your name and GT number on each page.**
2. The test is **CLOSED BOOK** and **NOTES**.
3. Please provide the answers in the space provided. You can use scratch paper (provided by us) to figure things out (if needed) but you get credit **only** for what you put down in the space provided for each answer.
4. For conceptual questions, **concise bullets** (**not wordy sentences**) are preferred.
5. Where appropriate use figures to convey your points (a figure is worth a thousand words!)
6. **Illegible answers are wrong answers.**
7. Please look through the whole test before starting so that you can manage your time better.

Good luck!

Question number	Points earned	Running total
1 (0 min) (Max: 1 pts)		
2 (12 min) (Max: 24 pts)		
3 (8 min) (Max: 16 pts)		
4 (15 min) (Max: 30 pts)		
5 (10 min) (Max: 20 pts)		
6 (5 min) (Max: 9 pts)		
Total (50 min) (Max: 100 pts)		

1. (0 min, 1 point)
The 2013 OSCAR for the best picture of the year goes to
(don’t worry…you get a point regardless of your answer)

[✓ a. Argo]
b. Life of Pi
c. Lincoln
d. Zero Dark Thirty
e. Django Unchained
f. None of the above
g. How I do know? I was studying for this exam…!

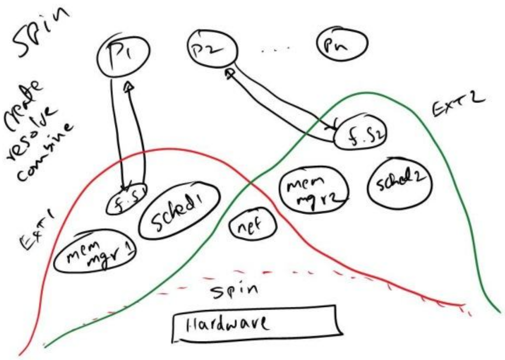
CS 6210 Spring 2013 Midterm Solution

Name:_______Kishore________________________GT Number:

OS Structures

2. (**12 min**, 24 points)

(a) (5 points) Draw a figure to show how SPIN supports multiple library operating systems on top of it.



1. Ext1 and Ext2 are two different library OSes that have been extended from basic SPIN.

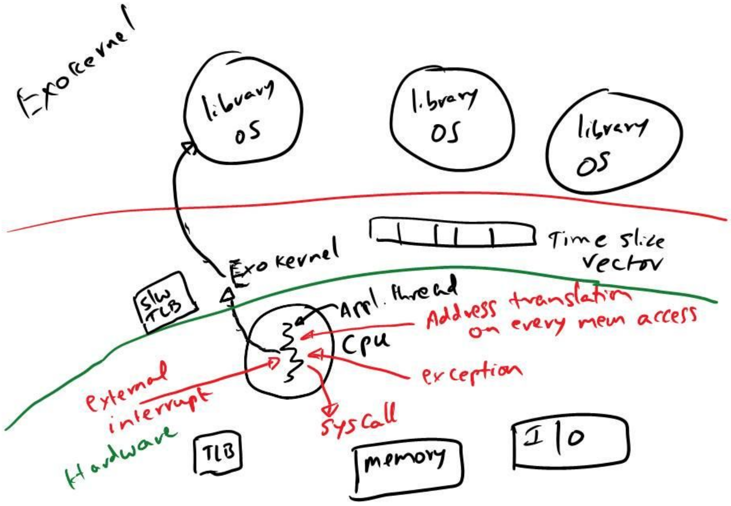
-2 if minor error
-4 if conceptual mistake

(b) (5 points) All the subsystems in the L3 microkernel live in user space. Every such subsystem has its own address space. The address space mechanism “grant” allows an address space to give a physical memory page frame to a peer subsystem. Can this compromise the “integrity” of a subsystem? Explain why or why not.
(**concise bullets please)**

- **No. All such mechanisms use the IPC of L3 which is one of the foundations for independence and integrity in L3.**
- **A subsystem that receives the memory page frame (via the grant mechanism) from a peer has the ability to “accept” or “reject” the page frame if it does not “trust” its peer.**

-2 if IPC not mentioned
-2 if “reject” not mentioned

(c) (10 points) This question is with reference to Exokernel. A user process makes a **system call**. Enumerate the sequence of steps (**with a figure** and **concise bullets**) by which this system call gets serviced by the library OS that this process belongs to.



Note: The figure can be much simpler than this! I am just being lazy and reproducing one of the slides from my lecture…

+1 for each bullet below
+5 for figure

- **the "syscall" results in a trap fielded by Exokernel**
- **Exokernel knows the "processor environment" (PE) data structures that corresponds to the currently executing library OS.**
- **Specifically, the "protected entry context" in the PE gives the entry point address in the library OS that corresponds to the syscall.**
- **Using this entry point, exokernel upcalls into the library OS.**
- **Library OS then completes the syscall**

(d) (4 points) Are the terms "protection domain" and "hardware address space" synonymous? If not, explain the difference between the two. (**concise bullets please)**

- **No. Protection domain is a logical concept that signifies the reach of a particular subsystem. For example, each specific service of an OS such as the memory manager, the file system, the network protocol stack, and the scheduler may each live in their own protection domains and interact with one another through well-defined interfaces implemented using IPC mechanism (*a la* L3 microkernel).**
- **Hardware address space is a physical concept. It pertains to the ability of the hardware to restrict the memory reach of the currently executing process. The OS can use this hardware mechanism to implement logical protection domains. A specific implementation of an OS may choose to pack multiple protection domains in the same hardware**

address space. Another may choose to associate a distinct hardware address space to each protection domain.

+2 for each point above

3. (**8 min**, 16 points)

(a) (8 points) Explain how content-based physical page sharing works across VMs in VMWare's ESX server (**figure and/or concise bullets please**)

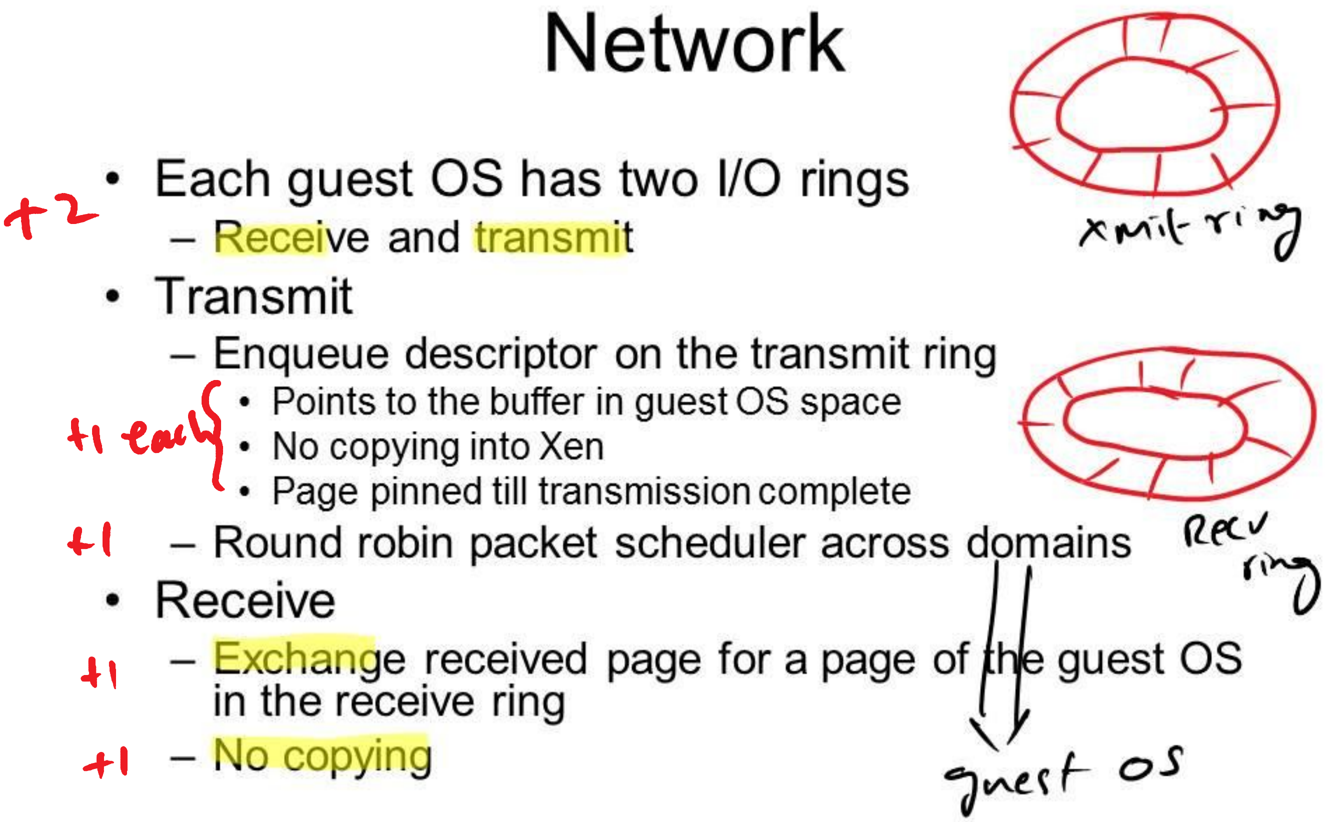
- **Scan a candidate PPN and generate a content hash**
- **If the hash matches an entry in the hash table, then perform full comparison of the scanned page and the matched page**
- **If the two pages are identical then mark both PPNs to point to the same MPN thus freeing up a machine page; mark both the PTEs in the respective shadow page tables (PPN->MPN entries) as "copy on write"**
- **If there is no match to the scanned page, then create a new "hint frame" entry in the hash table**
- **The scanning is done in the background during idle processor cycles**

+2 for each bullet (max +8)

CS 6210 Spring 2013 Midterm Solution

Name:________Kishore__________________________GT Number:

(b) (8 points) Explain how I/O rings may be used to implement the networking subsystem of a guest OS in Xen (use **pictures** and **concise bullets**)



Network

- Each guest OS has two I/O rings
  - Receive and transmit
- Transmit
  - Enqueue descriptor on the transmit ring
    - Points to the buffer in guest OS space
    - No copying into Xen
    - Page pinned till transmission complete
  - Round robin packet scheduler across domains
- Receive
  - Exchange received page for a page of the guest OS in the receive ring
  - No copying

Note: A much more succinct answer is acceptable. I am just being lazy and simply using my lecture slide!

+2 for picture
+4 for describing the transmit function
+2 for describing the receive function

CS 6210 Spring 2013 Midterm Solution

Name:_______Kishore_________________________GT Number:

<u>Synchronization and Communication in Parallel Systems</u>

4. (**15 mins**, 30 points)

(a) (10 points) In LRPC, the “client thread is **doctored** by the kernel to start executing the server code in the server domain.” Explain. (**concise bullets please**)

- +2 **The BO presented by the client code allows the kernel to identify the PDL associated with this call.**
- +2 **From the PDL, the kernel extracts the location of the E-stack that is to be used for this call.**
- **The kernel sets the SP to point to the E-stack**
- +2 **The kernel sets the address space pointer in the processor to point to the server’s address space**
- +2 **From the PDL, the kernel extracts the entry point address (in the server) for this call.**
- **The kernel sets the PC to point to this entry point address.**
- +2 **The “doctoring” is complete and the original client thread starts executing in the server address space at the entry point for this call using the E-stack as the stack.**

(b) (10 points) From the class discussions and the papers we have read on **processor scheduling** for shared memory multiprocessors, what are important things to keep in mind to ensure good performance? (**concise bullets please**)

- **Pay attention to cache affinity for a candidate process in selecting a processor to resume the candidate process**
- **Pay attention to the number of intervening processes that executed on the same processor since the time this candidate process ran on that processor**
- **Pay attention to the size of the scheduling queue on the processor before making a decision to place a candidate process on a processor queue**
- **In a multicore processor, misses in the LLC is expensive. Therefore, pay attention to the working set size of a thread in deciding the mix of threads that can be scheduled to run simultaneously on the multiple cores of the processor.**
- **The goal should be to ensure that the cumulative working set sizes of the co-scheduled threads can fit in the LLC to minimize misses and thus having to go off-chip.**

+2 for each bullet

Name:________Kishore___________________________GT Number:

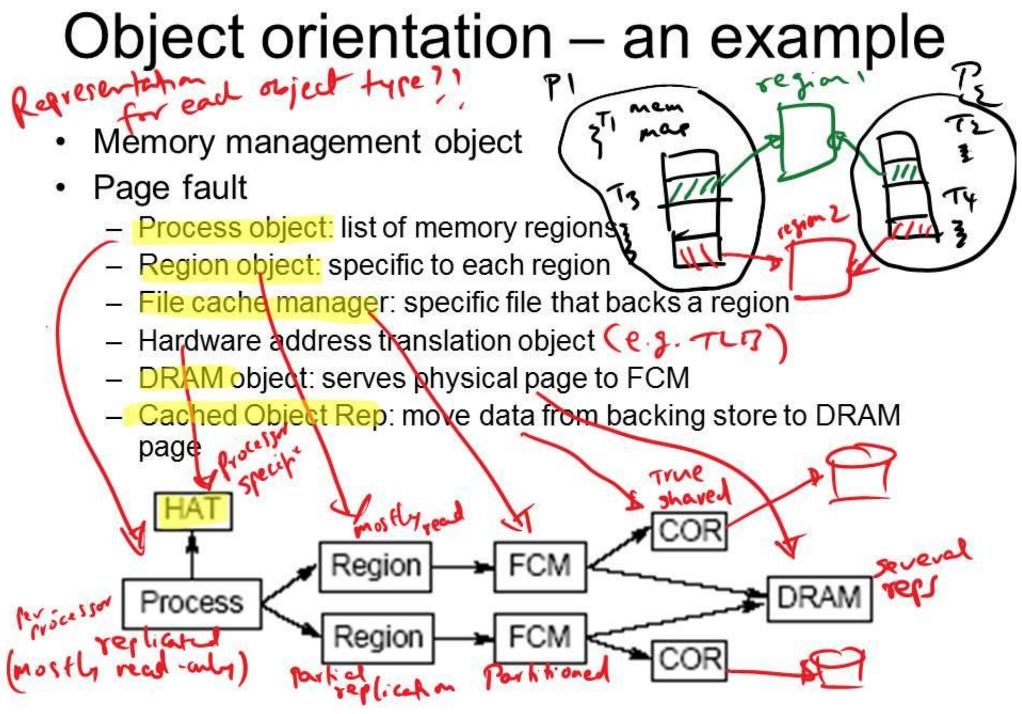
(c) (10 points)

(i) (2 points) What is the **thesis** of the Tornado approach to designing OS for shared memory multiprocessors?

- **To ensure scalability it is important to reduce the number of shared data structures in the kernel for which mutually exclusive access is necessary.**

All or nothing

(ii) (8 points) Show how this thesis can be applied to designing the memory management subsystem



Object orientation – an example

- Memory management object
- Page fault
  - Process object: list of memory regions
  - Region object: specific to each region
  - File cache manager: specific file that backs a region
  - Hardware address translation object (e.g. TLB)
  - DRAM object: serves physical page to FCM
  - Cached Object Rep: move data from backing store to DRAM page

+3 for figure

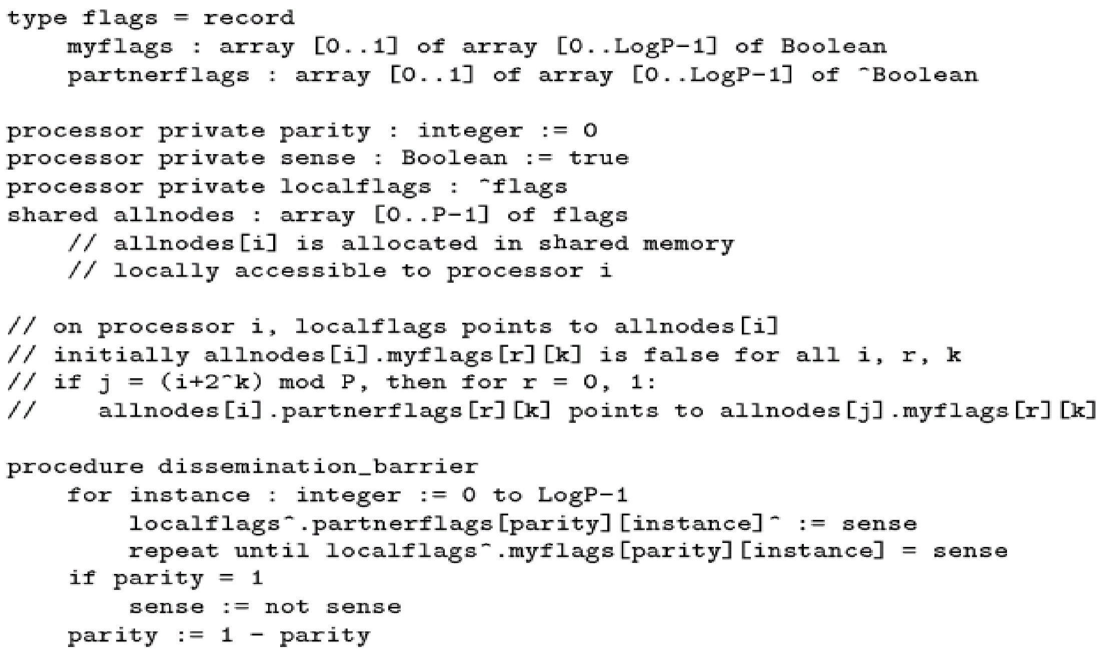
HAT: processor specific
Process: one rep per processor shared by all the threads (mostly read-only)
FCM: partitioned rep, region specific
Region: partial replication for a group of processors, since in the critical path of page faults (granularity decides concurrency for p.f. handling)
COR: a true shared object with a single rep since you don't except too much VM initiated file I/O with good mem mgmt (i.e., no thrashing)
DRAM: several representations

+1 for each

CS 6210 Spring 2013 Midterm Solution

Name:_______Kishore_______________________GT Number:

5. (**10 mins**, 20 points) Using a series of pictures, explain how barrier synchronization is achieved using the “Dissemination barrier” for **5 processors**. Just to jog your memory here is the code for the dissemination barrier.



```
type flags = record
    myflags : array [0..1] of array [0..LogP-1] of Boolean
    partnerflags : array [0..1] of array [0..LogP-1] of ^Boolean

processor private parity : integer := 0
processor private sense : Boolean := true
processor private localflags : ^flags
shared allnodes : array [0..P-1] of flags
    // allnodes[i] is allocated in shared memory
    // locally accessible to processor i

// on processor i, localflags points to allnodes[i]
// initially allnodes[i].myflags[r][k] is false for all i, r, k
// if j = (i+2^k) mod P, then for r = 0, 1:
//    allnodes[i].partnerflags[r][k] points to allnodes[j].myflags[r][k]

procedure dissemination_barrier
    for instance : integer := 0 to LogP-1
        localflags^.partnerflags[parity][instance]^ := sense
        repeat until localflags^.myflags[parity][instance] = sense
    if parity = 1
        sense := not sense
    parity := 1 - parity
```

Algorithm 9: The scalable, distributed dissemination barrier with only local spinning.

Use the next blank page for your answer.

Space for answer to Q5

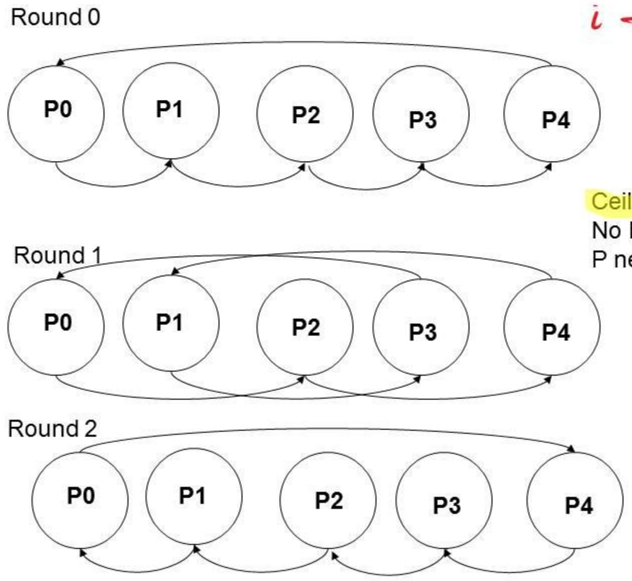
No hierarchy!

Dissemination barrier

round number

$i \longrightarrow (i+2^k) \bmod P$ in each round

(+3)



(+5)

(+5)

(+5)

(+2)

max number of rounds

Ceil (log P) rounds total

No longer pair-wise sync

P need not a perfect power of 2

especially useful in MP + NCC machines

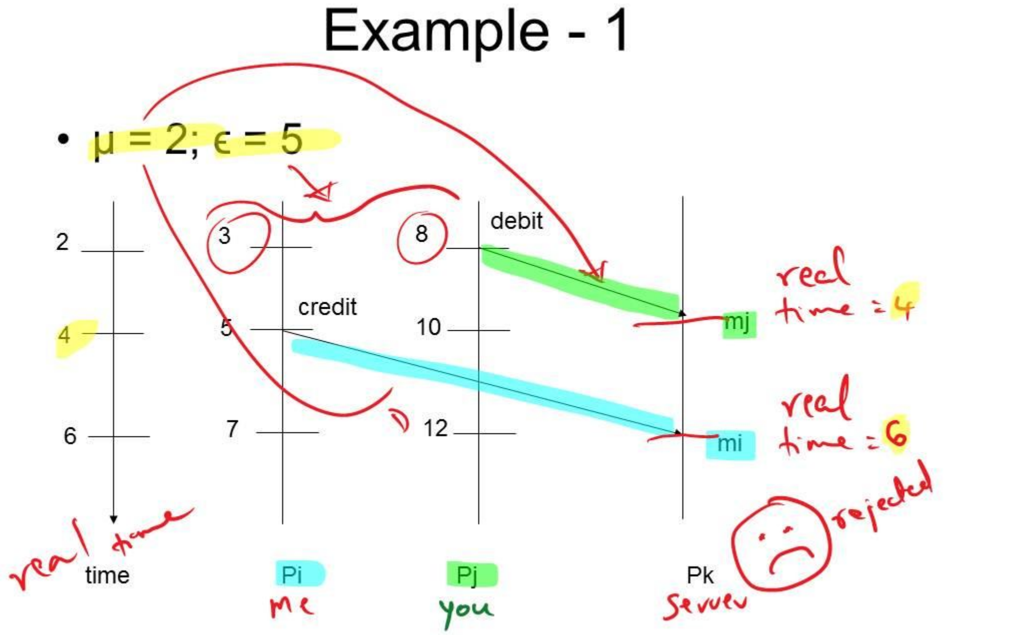
O(P) comm. events in each round

CS 6210 Spring 2013 Midterm Solution

Name:_______Kishore_________________________GT Number:

Communication in Distributed Systems

6. (**5 mins**, 9 points) Ignoring individual clock drift, show (with figures) how **IPC time** (**μ**) and mutual clock drift (**ϵ**) can lead to anomalies in a distributed system.



Example - 1

- $\mu = 2; \epsilon = 5$

Partial credit depends on specifics provided

+5 for knowing μ and ϵ relationships for anomaly
+4 for parts thereof for actually demonstrating an anomaly