## 1. Logical and Physical Address Spaces

The Kiwi™ memory architecture design team has a dilemma. The team is considering several different memory configuration variations for an upcoming machine design. Consider the following designs (All memory accesses are in terms of bytes, and all are using paging techniques):

| **Characteristic** | **Design 1** | **Design 2** | **Design 3** |
|---|---|---|---|
| Physical Memory Address Width | 8 bit | 16 bit | 32 bit |
| Logical Address Width | 12 bit | 20 bit | 24 bit |
| Page/Frame size in bytes | 16 bytes | 32 bytes | 64 bytes |
| Page Table Type | Single | Single | Double |

a) For each design, list the maximum number of pages each process can access in logical address space.
b) For each design, list the maximum number of frames in physical memory.
c) For design 3, if the outermost page table holds 32 entries, how many bits are needed in the logical address to represent the outer page table? How many bits are used for representing the offset within a page? How many bits are needed in the logical address in order to represent the inner page table?

## 2. Page Replacement

Given the following reference string:

0 2 1 3 0 1 4 0 1 2 3 4

show
a) Page faults occur during the processing of the reference scheme?
b) The ***hit ratio*** is for each of the following policies in a pure demand paging system?
c) What do you observe when you move from Scheme 1 to Scheme 2? Explain.

Scheme 1: FIFO with three pages of main memory
Scheme 2: FIFO with four pages of main memory

## 3. File System Implementation

Suppose a file system is constructed using blocks of 8 words each. In this system, a word has a length of 4 bytes. The disk pack used to hold the file system consists of 32 blocks. The initial block (block 0) contains a directory entry. The directory entry contains the filename of a single file in this file system, and a pointer to the first I-node in block 1. The I-node structure is as follows (word, value):

| | |
|---|---|
| 0 | Permission word |
| 1 | File Size |
| 2 | Direct block |
| 3 | Direct block |
| 4 | Direct block |
| 5 | Direct block |
| 6 | Single-indirect |
| 7 | Double-indirect |

The file contains 16 words of data: the first direct index points at block 31, and the second direct index points at block 29. Blocks 4,7,10,and 15 are marked bad. Assume that free blocks are allocated in logical order starting with block 0 and that write operations modify the file system 1 block at a time.

What will the state of the system look like after 100 additional words are appended to the file (draw a block diagram showing the structure of the I-node and the blocks that are allocated)

**4. Nachos Operating Systems**

1. List the two main operations (methods) that have the potential for resulting in a page fault.
2. Where is the address that caused the page fault stored?
3. You incremented the PC after every system call you implemented? Why not for page fault system call?
4. What is the purpose of Exec, Join and Exit system calls?
5. Explain the above with an example.
6. How would you control the access to a shared resource by multiple threads?

**5. Unix File system**

Consider the organization of a Unix file a represented by Inode. Assume that there 12 direct block pointers, and a singly, doubly and triply indirect pointers in each Inode. Assume that the system block size is 8K. Disk block pointer is 32 bits.

a. What is the maximum file size supported by the system?

b. Assuming no information other than the file Inode is in the main memory, how many disk accesses are required to access the byte in position 23,423,956.