CS 4348 Assignment 4

Daniel Crawford

1) Discuss two problems in fixed partitioning.

The two major problems with fixed partitioning can be narrowed down to inefficient use of memory due to internal fragmentation and the maximum number of processes is fixed. Respectively first, this is an inefficient way of handling memory because it does not allocate memory well to each process. This is because a process could be of only 1 MB, but the fixed size could be set at 8MB. This means that despite the program being much smaller, its still wasting lots of space. Next, the number of active processes is limited by this same logic. If each process over allocates memory, less processes can exist at the same time.

2) Explain how relative addressing supports relocation.

First of all, a relative address is basically a logical address, however it is expressed as a location relative to some known point. Specifically, this means a relative address holds an address that is used to be added with some register usually which provides the physical address. Relocation means that a process may take on different partitions over its lifespan. This means that the process can be swapped out at any point in time and be replaced. The process is not guaranteed to be given that specific partition once it returns. Relative addressing supports this relocation for reasons that fit well with these ideas. In a physical address, a process would be directly mapped to a partition but it could not be changed dynamically. With relative addressing, the partition which a process will lie will change depending on the base register which changed the physical address result from adding the base register and relative address. This means it can initially allocate a relative address to a space, and then change it later by simply setting a new value in the base register. Therefore, relative addressing supports relocation by adding a way to dynamically choose a physical address.

3) Given a 32 Megabyte memory space, illustrate the memory configuration after the each of the following requests using the Buddy System. Use a rule of first allocating the left block of a split pair or when more than one block of the same size is available.

**Bold: in use**

[---------------------------------------------32M--------------------------------------------]

P1: Requests 2M [**2M |** 2M |----4M----|---------8M---------|-----------------16M------------------------]

P2:Requests 4M [**2M |** 2M |----**4M**----|---------8M---------|-----------------16M------------------------]

P3:Requests 4M [**2M |** 2M |----**4M**----|---------**8M**---------|-----------------16M------------------------]

P4:Requests 4M [**2M |** 2M |----**4M**----|---------**8M**---------|----**4M----**|----4M----|---------8M--------]

P1:Releases 4M [---4M**----**|----**4M**----|---------**8M**---------|----**4M----**|----4M----|---------8M--------]