

Homework (Simulation)

This program allows you to see how address translations are performed in a system with segmentation. See the README for details.

Questions

1. First let's use a tiny address space to translate some addresses. Here's a simple set of parameters with a few different random seeds; can you translate the addresses?

```
segmentation.py -a 128 -p 512 -b 0 -l 20 -B 512
-L 20 -s 0
segmentation.py -a 128 -p 512 -b 0 -l 20 -B 512
-L 20 -s 1
segmentation.py -a 128 -p 512 -b 0 -l 20 -B 512
-L 20 -s 2
```

2. Now, let's see if we understand this tiny address space we've constructed (using the parameters from the question above). What is the highest legal virtual address in segment 0? What about the lowest legal virtual address in segment 1? What are the lowest and highest *illegal* addresses in this entire address space? Finally, how would you run `segmentation.py` with the `-A` flag to test if you are right?
3. Let's say we have a tiny 16-byte address space in a 128-byte physical memory. What base and bounds would you set up so as to get the simulator to generate the following translation results for the specified address stream: valid, valid, violation, ..., violation, valid, valid? Assume the following parameters:

```
segmentation.py -a 16 -p 128
-A 0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15
--b0 ? --l0 ? --b1 ? --l1 ?
```

4. Assume we want to generate a problem where roughly 90% of the randomly-generated virtual addresses are valid (not segmentation violations). How should you configure the simulator to do so? Which parameters are important to getting this outcome?
5. Can you run the simulator such that no virtual addresses are valid? How?