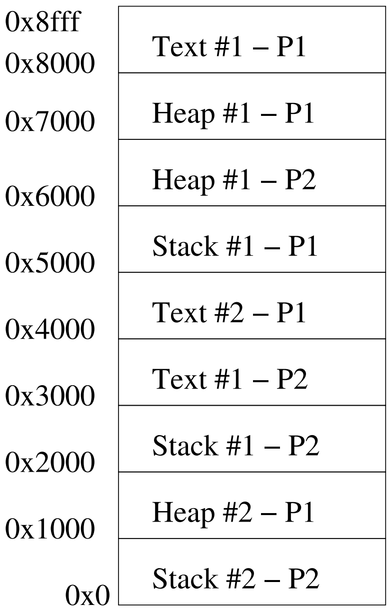
Question

(The errors in the original quiz have been corrected below.)

A computer has 2 processes (P1 and P2) loaded into RAM (ignore the kernel for this question). Assume the RAM is divided into areas, each 4KB in size, and allocated to the 2 processes as shown in the figure. The architecture uses a (tiny) 64KB address space, and expects the text to be at the bottom (staring address 0), the heap immediately above the text, and the stack to grow down from the top (largest address in the stack is `0xffff`).

Give the mapping functions for P1 and P2 as mathematical functions.



Answer

(I've only drawn up the P1 mapping function. P2's is very similar, but with 8KB of heap area and 8KB of stack, and with the pointers going to different physical addresses).

The mapping function takes an address from a process (often called a virtual address) and returns a physical address. By adding this level of indirection, the operating system (kernel) can control - by remapping - the physical addresses that any process can name (and, if a process cannot name something, it can't use it).

There are many ways to represent a mathematical function. The way I've chosen is to simply draw the mapping from its domain to its range (the diagram may be a little hard to read; note that the domain - the virtual addresses - are on the right).

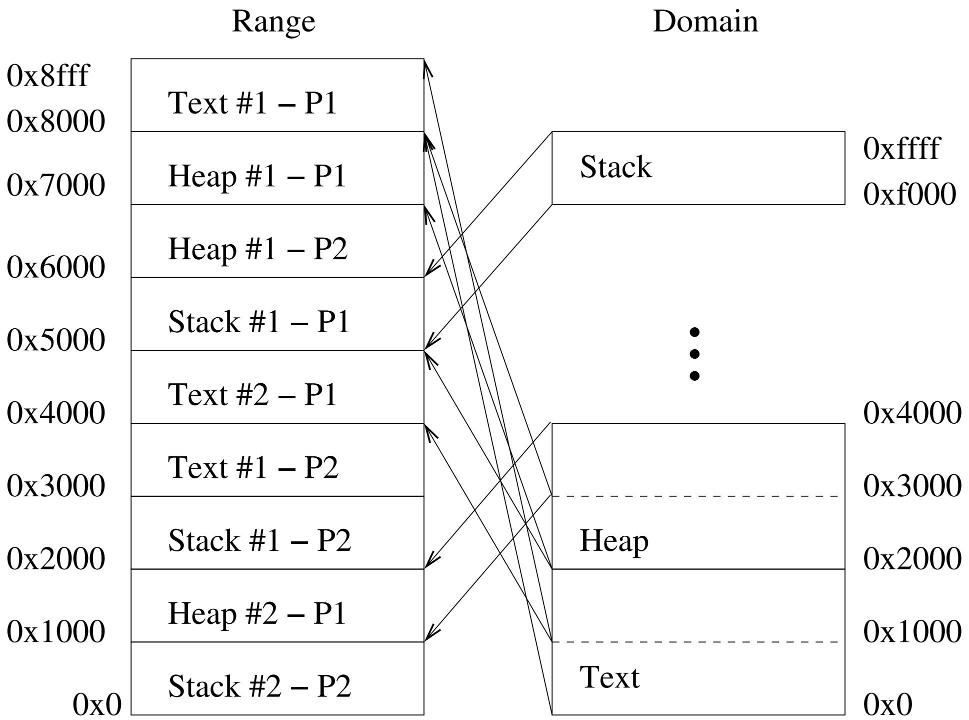


Figure 1: Mapping Function for P1

As an alternative, one might use a multi-part function:

$$mapping_{P1}(vaddr) = \begin{cases} \texttt{0x8000} + vaddr & if\ vaddr \geq 0\ and\ vaddr < \texttt{0x1000} \\ \texttt{0x4000} + (vaddr - \texttt{0x1000}) & if\ vaddr \geq \texttt{0x1000}\ and\ vaddr < \texttt{0x2000} \\ \texttt{0x7000} + (vaddr - \texttt{0x2000}) & if\ vaddr \geq \texttt{0x2000}\ and\ vaddr < \texttt{0x3000} \\ \texttt{0x1000} + (vaddr - \texttt{0x3000}) & if\ vaddr \geq \texttt{0x3000}\ and\ vaddr < \texttt{0x4000} \\ \texttt{0x5000} + (vaddr - \texttt{0xf000}) & if\ vaddr \geq \texttt{0xf000}\ and\ vaddr < \texttt{0x10000} \end{cases}$$

Note that, though the stack is growing down, the mapping function doesn't really need to know this.

We'll talk about how this mapping function might be implemented in a few weeks.