Code from vm_fault

```
vbase1 = as->as_vbase1;
vtop1 = vbase1 + as->as_npages1 * PAGE_SIZE;
vbase2 = as->as_vbase2;
vtop2 = vbase2 + as->as_npages2 * PAGE_SIZE;
stackbase = USERSTACK - DUMBVM_STACKPAGES * PAGE_SIZE;
stacktop = USERSTACK;

if (faultaddress >= vbase1 && faultaddress < vtop1) {  code
   paddr = (faultaddress - vbase1) + as->as_pbase1;
}
else if (faultaddress >= vbase2 && faultaddress < vtop2) {   data
   paddr = (faultaddress - vbase2) + as->as_pbase2;
}
else if (faultaddress >= stackbase && faultaddress < stacktop) {  stack
   paddr = (faultaddress - stackbase) + as->as_stackpbase;
}
else {
   return EFAULT;
}
```

Field	Process 1
as_vbase1	0x0040 0000
as_pbase1	0x0020 0000
as_npages1	0x0000 0008
as_vbase2	0x1000 0000
as_pbase2	0x0080 0000
as_npages2	0x0000 0010
as_stackpbase	0x0010 0000

Code from vm_fault

```
vbase1 = as->as_vbase1;   (0x0040 0000)
vtop1 = vbase1 + as->as_npages1 * PAGE_SIZE; (0x0040 8000)
vbase2 = as->as_vbase2;   (0x1000 0000)
vtop2 = vbase2 + as->as_npages2 * PAGE_SIZE; (0x1001 0000)
stackbase = USERSTACK - DUMBVM_STACKPAGES * PAGE_SIZE;
            (0x8000 0000 – 12 * 0x1000 = 0x7FFF 4000)
stacktop = USERSTACK;  (0x8000 0000)

if (faultaddress >= vbase1 && faultaddress < vtop1) {
   paddr = (faultaddress - vbase1) + as->as_pbase1;
}
else if (faultaddress >= vbase2 && faultaddress < vtop2) {
   paddr = (faultaddress - vbase2) + as->as_pbase2;
}
else if (faultaddress >= stackbase && faultaddress < stacktop) {
   paddr = (faultaddress - stackbase) + as->as_stackpbase;
}
else {
   return EFAULT;
}
```

Field	Process 1
as_vbase1	0x0040 0000
as_pbase1	0x0020 0000
as_npages1	0x0000 0008
as_vbase2	0x1000 0000
as_pbase2	0x0080 0000
as_npages2	0x0000 0010
as_stackpbase	0x0010 0000

Code from vm_fault

```
vbase1 = as->as_vbase1;   (0x0040 0000)
vtop1 = vbase1 + as->as_npages1 * PAGE_SIZE; (0x0040 8000)
vbase2 = as->as_vbase2;   (0x1000 0000)
vtop2 = vbase2 + as->as_npages2 * PAGE_SIZE; (0x1001 0000)
stackbase = USERSTACK - DUMBVM_STACKPAGES * PAGE_SIZE;
                (0x8000 0000 – 12 * 0x1000 = 0x7FFF 4000)
stacktop = USERSTACK;  (0x8000 0000)

if (faultaddress >= vbase1 && faultaddress < vtop1) {
   paddr = (faultaddress - vbase1) + as->as_pbase1;
}
else if (faultaddress >= vbase2 && faultaddress < vtop2) {
   paddr = (faultaddress - vbase2) + as->as_pbase2;
}
else if (faultaddress >= stackbase && faultaddress < stacktop) {
   paddr = (faultaddress - stackbase) + as->as_stackpbase;
}
else {
   return EFAULT;
}
```

Field	Process 1
as_vbase1	0x0040 0000
as_pbase1	0x0020 0000
as_npages1	0x0000 0008
as_vbase2	0x1000 0000
as_pbase2	0x0080 0000
as_npages2	0x0000 0010
as_stackpbase	0x0010 0000

Virtual Address = 0x0040 0004

In segment 1 (code) because the address is in between vbase1 and vtop1

Physical Address = 0x0040 0004 - 0x0040 0000 + 0x0020 0000
= 0x0020 0004

Code from vm_fault

```
vbase1 = as->as_vbase1;   (0x0040 0000)
vtop1 = vbase1 + as->as_npages1 * PAGE_SIZE; (0x0040 8000)
vbase2 = as->as_vbase2;   (0x1000 0000)
vtop2 = vbase2 + as->as_npages2 * PAGE_SIZE; (0x1001 0000)
stackbase = USERSTACK - DUMBVM_STACKPAGES * PAGE_SIZE;
            (0x8000 0000 – 12 * 0x1000 = 0x7FFF 4000)
stacktop = USERSTACK;  (0x8000 0000)

if (faultaddress >= vbase1 && faultaddress < vtop1) {
   paddr = (faultaddress - vbase1) + as->as_pbase1;
}
else if (faultaddress >= vbase2 && faultaddress < vtop2) {
   paddr = (faultaddress - vbase2) + as->as_pbase2;
}
else if (faultaddress >= stackbase && faultaddress < stacktop) {
   paddr = (faultaddress - stackbase) + as->as_stackpbase;
}
else {
   return EFAULT;
}
```

Field	Process 1
as_vbase1	0x0040 0000
as_pbase1	0x0020 0000
as_npages1	0x0000 0008
as_vbase2	0x1000 0000
as_pbase2	0x0080 0000
as_npages2	0x0000 0010
as_stackpbase	0x0010 0000

Virtual Address = 0x1000 91A4

In segment 2 (data) because the address is in between vbase2 and vtop2

Physical Address = 0x1000 91A4 - 0x1000 0000 + 0x0080 0000
= 0x0080 91A4

Code from vm_fault

```
vbase1 = as->as_vbase1;   (0x0040 0000)
vtop1 = vbase1 + as->as_npages1 * PAGE_SIZE; (0x0040 8000)
vbase2 = as->as_vbase2;   (0x1000 0000)
vtop2 = vbase2 + as->as_npages2 * PAGE_SIZE; (0x1001 0000)
stackbase = USERSTACK - DUMBVM_STACKPAGES * PAGE_SIZE;
                (0x8000 0000 – 12 * 0x1000 = 0x7FFF 4000)
stacktop = USERSTACK;  (0x8000 0000)

if (faultaddress >= vbase1 && faultaddress < vtop1) {
   paddr = (faultaddress - vbase1) + as->as_pbase1;
}
else if (faultaddress >= vbase2 && faultaddress < vtop2) {
   paddr = (faultaddress - vbase2) + as->as_pbase2;
}
else if (faultaddress >= stackbase && faultaddress < stacktop) {
   paddr = (faultaddress - stackbase) + as->as_stackpbase;
}
else {
   return EFAULT;
}
```

Field	Process 1
as_vbase1	0x0040 0000
as_pbase1	0x0020 0000
as_npages1	0x0000 0008
as_vbase2	0x1000 0000
as_pbase2	0x0080 0000
as_npages2	0x0000 0010
as_stackpbase	0x0010 0000

Virtual Address = 0x7FFF 41A4

In the stack because the address is in between stackbase and stacktop

Physical Address = 0x7FFF 41A4 - 0x7FFF 4000 + 0x0010 0000
= 0x0010 01A4

Code from vm_fault

```
vbase1 = as->as_vbase1;   (0x0040 0000)
vtop1 = vbase1 + as->as_npages1 * PAGE_SIZE; (0x0040 8000)
vbase2 = as->as_vbase2;   (0x1000 0000)
vtop2 = vbase2 + as->as_npages2 * PAGE_SIZE; (0x1001 0000)
stackbase = USERSTACK - DUMBVM_STACKPAGES * PAGE_SIZE;
                (0x8000 0000 – 12 * 0x1000 = 0x7FFF 4000)
stacktop = USERSTACK;  (0x8000 0000)

if (faultaddress >= vbase1 && faultaddress < vtop1) {
   paddr = (faultaddress - vbase1) + as->as_pbase1;
}
else if (faultaddress >= vbase2 && faultaddress < vtop2) {
   paddr = (faultaddress - vbase2) + as->as_pbase2;
}
else if (faultaddress >= stackbase && faultaddress < stacktop) {
   paddr = (faultaddress - stackbase) + as->as_stackpbase;
}
else {
   return EFAULT;
}
```

Field	Process 1
as_vbase1	0x0040 0000
as_pbase1	0x0020 0000
as_npages1	0x0000 0008
as_vbase2	0x1000 0000
as_pbase2	0x0080 0000
as_npages2	0x0000 0010
as_stackpbase	0x0010 0000

Virtual Address = 0x7FFF 32B0

Not in any of the segments.
Throw an exception