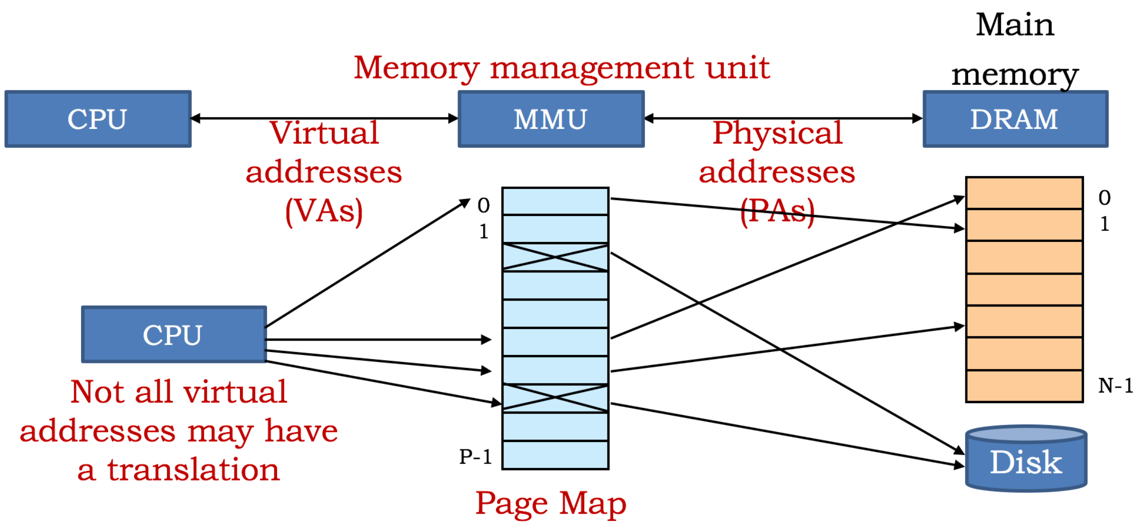
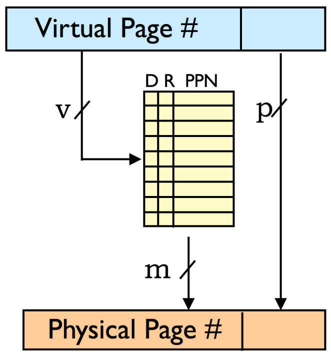
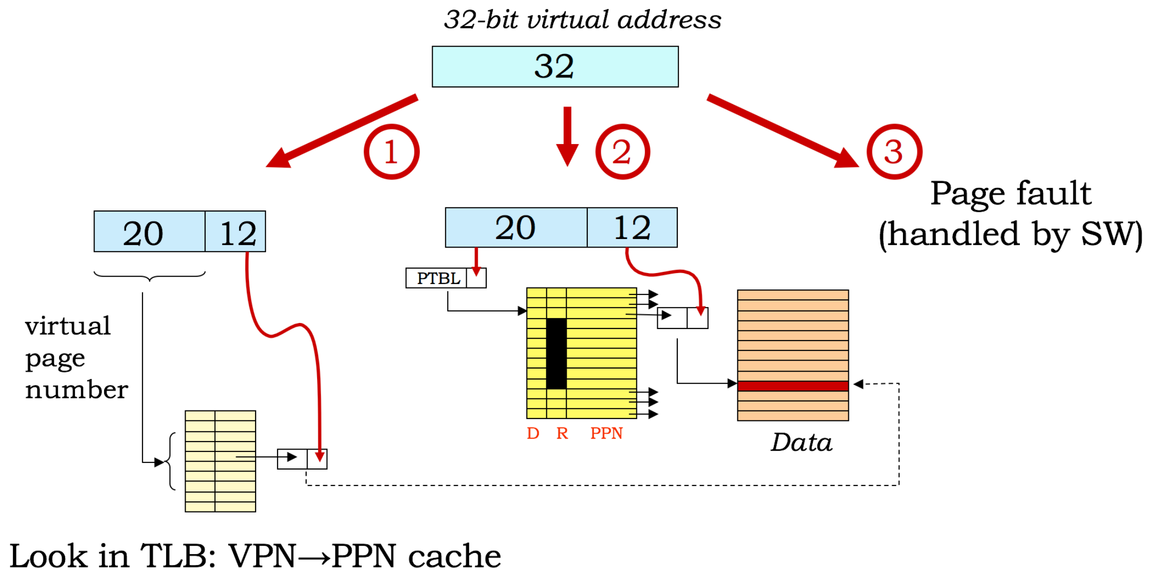
6.004 Tutorial Problems
L18 – Virtual Memory





$(v + p)$	bits in virtual address
$(m + p)$	bits in physical address
2^v	number of *virtual* pages
2^m	number of *physical* pages
2^p	bytes per physical page
2^{v+p}	bytes in virtual memory
2^{m+p}	bytes in physical memory
$(m+2)2^v$	bits in the page table



Note: A subset of problems are marked with a red star (★). We especially encourage you to try these out before recitation.

Problem 1. ★

The micro-RISC has a 12-bit virtual address, an 11-bit physical address and uses a page size of 256 (= 2^8) bytes. The micro-RISC has been running for a while and at the current time the page table has the contents shown on the right. Assume all physical pages are currently in use.

VPN	*D*	*R*	*PPN*
0	0	1	2
1	—	0	—
2	0	1	4
3	—	0	—
4	1	1	0
5	1	1	1
6	—1	0>1	—>5
7	—	0	—
8	—	0	—
9	—	0	—
A	—	0	—
B	—	0	—
C	1	1	7
D	1	1	6
LRU→E	1>-	1>0	5 > -
F	0	1	3

(A) Assuming each page table entry contains the usual dirty (D) and resident (R) bits, what it is the total size of the page table in bits

Size of page table (bits): ___**2^4*5=80**_______

Since each page is 2^8 bytes, the page offset is 8bits.
A 12-bit Virtual Address means the VPN is then 4bits
And an 11-bit Phyiscal address gives us a PPN of 3 bits/
Each table entry is then 2(Dirty, Resident)+3(PPN)=5bits

4-bit VPN means we have 2^4 entries, for 80 total bits

(B) The following load instruction, located at virtual address 0x0BC, is about to be executed.

```
lw x1, 0x2C8(x0)
```

```
Lower 8 bits are offset, VPN is bits 11:8
```

0x0BC => VPN=0 => PPN=2
0x2C8 => VPN=2 => PPN=4

Attach PPN back to offset

0x0BC => 0x2Bc
0x2C8 => 0x4C8

When the instruction is executed, what main memory locations are accessed by the instruction fetch and then the memory access initiated by the LW? Use the page table shown to the right. Assume the LRU page is virtual page 0xE.

Physical address for instruction fetch: 0x__2BC________

Physical addr for data read by LW instruction: 0x ___4C8_____

(C) A few instructions later, the following instruction, located at virtual address 0x0CC, is executed:

```
sw x1, 0(x2)    // current value of X2 = SP = 0x600
```

Please mark up the page table to show its contents after the SW has been executed. Use the page table shown to the right. Assume the LRU page is virtual page 0xE.

Remember to show any changes to the dirty and resident control bits as well as updates to the physical page numbers. If an entry in the page table no longer matters, please indicate that by replacing it with "— 0 — " for the D, R and PPN entries.

Show updated contents of page table

0x0cc => VPN=0 => PPN = 2

0x600 => VPN=6 => page fault, because nothing is resident. So, we have to evict the LRU page (E), which also means we have to write back PPN 5 to disk because D=1, dirty. Now, PPN 5 is available for VPN 6, and we set dirty to 1 because the instruction is a store instruction.

Problem 2.

Consider a RISC-V processor that includes a 40-bit virtual address, an MMU that supports 4096 (2^{12}) bytes per page, 2^{32} bytes of physical memory, and a large Flash memory that serves as a disk. The MMU and the page fault handler implement an LRU replacement strategy.

Note that in the RISC-V processor we have been building in class, the word size is 32 bits. In order to support a 40-bit virtual address space, this problem is referring to a RISC-V processor that uses a larger (>= 40 bit) word size.

(A) What is the size of the page table for this processor? Assuming the page table includes the standard dirty and resident bits, specify the width of each page table entry in bits, and number of entries in the page table.
2^12 bytes per page gives 12 offset bits. That means we have 28 bits left for the VPN in the 40 bit VA and 20 bits left in the 32 bit PA

Size of page table entry in bits:__ 20(PPN)+2(Dirty, Resident)=22________

Number of entries in the page table:____ 2^{28} (number of possible VPNs)_____

(B) The following test program is running on this RISC-V processor. The first 8 locations of the page table, just before executing this test program, are shown below; the least-recently-used page ("LRU") and next least-recently-used page ("next LRU") are as indicated. This RISC-V processor also has a 4 element, fully associative, Translation Lookaside Buffer (TLB) that caches page table translations from VPN to PPN.

```
. = 0x0
lui x3, 2
lw x5, 0x600(x3)
lui x3, 4
sw x5, 0x100(x3)
```

TLB

	Tag (VPN)	*D*	*R*	*PPN*
LRU→	0x3	1	1	0x1
	0x2	0	1	0x3
	0x6	0	1	0x2
Next LRU→	0x1	0	1	0x5

Page Table

	VPN	*D*	*R*	*PPN*
	0	1	1	0x7
	1	0	1	0x5
	2	0	1	0x3
write back to disk *LRU→*	*3*	1	1 0	0x1 -
	4	--1	0 1	--0x1
	5	0	1	0x0
	6	0	1	0x2
Next LRU→	*7*	0	1	0x6

For each virtual page that is accessed by this program, specify the **VPN**, whether or not it results in a **TLB hit** on the first access to that page, whether or not it results in a **page fault**, and the **PPN** that the page ultimately maps to. *You may not need to use all rows of the table*
The VPN is the upper 28 bits of each address.

VPN	**TLB Hit (Yes/No)**	**Page Fault (Yes/No)**	**PPN**
0	No	No	7
2	Yes	No	3
4	No	Yes	1

lw: 0x600+0x2000=0x2600=0b0010_0110_0000_0000, VPN=2
sw: 0x100+0x4000=0x4100, VPN=4

(C) Which physical pages, if any, need to be written to disk during the execution of the test program in part B?

Physical page numbers written to disk or NONE: ___1______

(D) What is the physical address of the LW instruction?

Physical address of LW instruction: 0x___7004______

The LW instruction is at address 0x4, giving VPN=0 which translates to PPN=7

Now reattach the page offset of 0x004 and we get 0x7004

Problem 3.

Consider a RISC-V processor that includes a 32-bit virtual address, an MMU that supports 4096 (2^{12}) bytes per page, 2^{24} bytes of physical memory, and a large Flash memory that serves as a disk. The MMU and the page fault handler implement an LRU replacement strategy.

(A) The designers are thinking about implementing the page table using a separate SRAM memory with L entries, where each entry has B bits. If the page table includes the standard dirty and resident bits, what are the appropriate values for the parameters L and B?

2^12 bytes per page gives 12 offset bits. That means we have 20 bits left for the VPN in the 32 bit VA and 12 bits left in the 24 bit PA

Appropriate value for the parameter L: ___2^{20} (number of possible VPNs)______

Appropriate value for the parameter B: __12(PPN)+2(Dirty, Resident)=14______

(B) If the designers decide to decrease the page size to 2048 (2^{11}) bytes but keep the same size virtual and physical addresses, what affect will the change have on the following architectural parameters? Use a letter "a" through "e" to indicate how the *new* value of the parameter compares to the *old* value of the parameter:

One less bit of page offset means 1 more bit for VPN and PPN

(a) doubled (b) increased by 1 (c) stays the same (d) decreased by 1 (e) halved

One more bit of PPN in the entry **Size of page table entry in bits:** _B__

One more VPN bit, double # of VPNs **Number of entries in the page table:** __A___

Maximum percentage of virtual memory that can be resident at any given time:___C__
$2^{24}/2^{11}=2^{13}$ physical pages
$2^{32}/2^{11}=2^{21}$ virtual pages
Old: $2^{12}/2^{20}=2^{-8}$, new: $2^{13}/2^{21}=2^{-8}$

(D) A test program has been running on the RISC-V with a page size of 2^{12} bytes and has been halted *just before* execution of the following instruction at location 0x1234. Assume the current contents of x2 are 0x3000.

```
    sw x1, 0x4C8(x2)    | PC = 0x1234
```

VA=0x34c8, VPN=3, not resident

The first 8 locations of the page table at the time execution was halted are shown to the right; the least-recently-used page ("LRU") and next least-recently-used page ("next LRU") are as indicated. Assume that all the pages in physical memory are in use. Execution resumes and the SW instruction is executed.

VPN	*D*	*R*	*PPN*
0	1	1	0x1
1	0	1	0x0
2	1	1	0x6
3	--1	0 1	--0x7
Next LRU→ 4	0	1	0x4
5	0	1	0x2
LRU→ 6	1 -	1 0	0x7 -
7	0	1	0x3

Please **show the contents of the page table** after the SW instruction has completed execution by crossing out any values that changed and writing in their new values. Note that the D and PPN fields for a non-resident page do not need to be specified.

(E) Which physical pages, if any, need to be written to disk during the execution of the SW instruction in part (D)?

Physical page numbers written to disk or NONE: __7___

Since the desired page is not resident, we have to evict the LRU page. That's ppn 0x7, and it happens to be dirty so we have to write it back to disk.

Problem 4. ★

Consider a virtual memory system that uses a single-level page table to translate virtual addresses into physical addresses. Each of the questions below asks you to consider what happens when **just ONE of the design parameters** (page size, virtual memory size, physical memory size) of the original system is changed. **Circle the correct answer.**

(A) If the physical memory size (in bytes) is **doubled**, the number of entries in the page table
(a) stays the same
(b) doubles
(c) is reduced by half
(d) increases by one
(e) decreases by one

# entries only depends on # bits of VPN

(B) If the page size (in bytes) is **halved**, the number of entries in the page table
(a) stays the same
(b) doubles
(c) is reduced by half
(d) increases by one
(e) decreases by one

VA same size, VPN is one bit larger

(C) If the virtual memory size (in bytes) is **doubled**, the number of bits in each entry of the page table
(a) stays the same
(b) doubles
(c) is reduced by half
(d) increases by one
(e) decreases by one

Entry only has PPN and dirty/resident bits

(D) If the page size (in bytes) is **doubled**, the number of bits in each page table entry
(a) stays the same
(b) doubles
(c) is reduced by half
(d) increases by one
(e) decreases by one

PPN is one bit smaller

Consider a virtual memory system for a new processor with 4096 (2^{12}) virtual pages and 16384 (2^{14}) physical pages where each page contains 1024 (2^{10}) bytes. The first 8 entries of the current page table are shown below:

index	**D**	**R**	**PPN**
0	1	1	0x22
1	0	1	0x01
2	--	0	--
3	0	1	0x02
4	1	1	0x03
5	--	0	--
6	1	1	0x15
7	0	1	
...			

(E) What is the total number of bits in the page table?

Total number of bits in the page table: __$2^{12}*16 = 2^{16}$____

2^12 Virtual pages/entries*(Dirty:1+Resident:1+PPN:14) = $2^{12}*16$

(F) Which address bits from the CPU are used to choose an entry from the page table?

Address bits used to choose page table entry: A[_21_ : _10_ **]**

We have a 10 bit page offset, so we start after the 10 lowest bits

(G) What is the physical address for the word at virtual location 0x1234? Write "not resident" if the location is not currently present in physical memory.

Physical address for byte at virtual address 0x1234 or "not resident": __0xE34____

0x1234=0001_00**10_0011_0100** (bold bits are offset)
VPN=4 translates to PPN=3, reattach offset
0000_11**10_0011_0100 = 0xE34**

(H) Briefly explain what action caused the D bit for page 6 to be 1.

A store instruction wrote to a location in virtual page 6 **Briefly explain.**

Problem 5.

(A) A particular RISC-V implementation has 32-bit virtual addresses, 32-bit physical addresses and a page size of 2^{12} bytes. A test program has been running on this RISC-V and has been halted *just before* execution of the following instruction at location 0x1FFC. Assume x2 = 0x3000 and x3 = 0x6000 just prior to executing these instructions.

12 bits of page offset, 20 bit VPN and PPN

```
lw x1, 0x4C8(x2)    | PC = 0x1FFC
sw x1, 0x4(x3)      | PC = 0x2000
```

```
Access 0x4C8+0x3000=0x34C8 for LW, 0x4+0x6000=0x6004 for SW

LW inst:   VA=1FFC, VPN=1, PPN=0
LW access: VA=34C8, VPN=3, miss, evict LRU from VPN 2, PPN=6

SW inst:   VA=2000, VPN=2, miss, evict next LRU VPN 4, PPN=4
SW access: VA=6004, VPN=6, PPN=7
```

The first 8 locations of the page table at the time execution was halted are shown below; the least recently used page ("LRU") and next least recently used page ("next LRU") are as indicated. Assume that all the pages in physical memory are in use. Execution resumes and **both** the LW and SW instructions are executed.

Please **show the contents of the page table** after the SW instruction has completed execution by crossing out any values that changed and writing in their new values.

VPN	*D*	*R*	*PPN*
0	1	1	0x1
1	0	1	0x0
LRU→ 2	1 0	1 0,1	0x6 0x4
3	--0	0 1	- 0x6
Next LRU→ 4	0 -	1 0	0x4 -
5	0	1	0x2
6	0 1	1	0x7
7	0	1	0x3

(B) Which physical pages, if any, needed to be written to disk during the execution of the LW and SW instructions?

Physical page numbers written to disk or NONE: ____6_____

PPN 6 had its dirty bit set when we evicted it, so we write it back to disk

(C) Please give the 32-bit physical memory addresses used for the four memory accesses associated with the execution of the LW and SW instruction.

32-bit physical memory address of LW instruction: 0x_____0FFC__________

32-bit physical memory address of data read by LW: 0x____64C8___________

32-bit physical memory address of SW instruction: 0x____4000________

32-bit physical memory address of data written by SW: 0x_____7004______

See previous page for explanations

Problem 6. ★

Consider a system with 40-bit virtual addresses, 36-bit physical addresses, and 64 KB (2^{16} bytes) pages. The system uses a page table to translate virtual addresses to physical addresses; each page table entry include dirty (D) and resident (R) bits.

Note that in the RISC-V processor we have been building in class, the word size is 32 bits. In order to support a 40-bit virtual address space, this problem is referring to a processor that uses a larger (>= 40 bit) word size.

16 bit offset, 40-16=24 bit VPN, 36-16=20 bit PPN

(A) (2 points) Assuming a flat page table, what is the size of each page table entry, and how many entries does the page table have?

Size of page table entry in bits: ____20(PPN)+2(Dirty, Resident)=22________

Number of entries in the page table: ____2^{24} (number of possible VPNs)______

(B) (1 point) If you changed the system to use 16 KB (2^{14} bytes) pages instead of 64 KB pages, how would the number of entries in the page table change? Please give the ratio of the new size to the old size.

2 less bits of offset means 2 more bits of VPN and PPN. 2 more VPN bits means 4x the number of page table entries

(# entries with 16 KB pages) / (# entries with 64 KB pages): ____4________

Assume 64 KB pages for the rest of this exercise.

(C) (6 points) The contents of the page table and TLB are shown to the right. The page table uses an LRU replacement policy, and the LRU page (shown below) will be chosen for replacement. For each of these four accesses, compute its corresponding physical address and indicate whether the access causes a TLB miss and/or a page fault. **Assume each access starts with the TLB and Page Table state shown to the right.**

TLB

VPN (tag)	V	D	PPN
0x0	1	0	0xBE7A
0x3	0	0	0x7
0x5	1	1	0xFF
0x2	1	0	0x900

VPNs in red

Fill in table below

	Virt Addr	**PPN (in hex)**	**Phys Addr (in hex)**	**TLB Miss?**	**Page Fault?**
1.	**0x06004**	_0xBE7A_	__**0xBE7A6004**__	**Y** / [✓ N]	**Y** / [✓ N]
2.	**0x30286**	__0x8__	_**0x80286**_	[✓ Y] / **N**	**Y** / [✓ N]
3.	**0x68030**	_0x70__	_**0x708030**_	[✓ Y] / **N**	**Y** / [✓ N]
4.	**0x4BEEF**	_0x8_	__**0x8BEEF**_	[✓ Y] / **N**	[✓ Y] / **N**

Page Table

VPN	R	D	PPN
0	1	0	0xBE7A
1	0	0	---
2	1	0	0x900
3	1 0	0	0x8
4	0 1	0	--- 0x8
5	1	1	0xFF
6	1	0	0x70
...			

LRU PAGE ← (VPN 3)

Problem 7. (SP'20 Quiz 3 Problem 3)

For the following questions, assume a processor with 64-bit virtual addresses, 40-bit physical addresses and page size of 4096 (2^{12}) bytes per page. The Page Table of this processor uses an LRU replacement strategy, and handles missing pages using a page fault handler.

A) What is the size of the page table? Assume that each page table entry includes a **dirty bit** and a **resident bit**. Specify the number of page table entries and the width of each entry.

Number of entries in the page table: ____2^52__

Width of each page table entry (bits): _____30___

B) What is the maximum fraction of virtual memory that can be resident in physical memory at any given time (assuming the page table is not in physical memory)?

Max fraction of virtual memory that can be resident in physical memory: _____1/2^24_____

C) If we half the size of virtual memory but keep the same physical address length and page size (2^{12} bytes per page), what effect will the change have on the size of a page table entry and on the number of entries in the page table? Use a letter "a" through "e" to indicate how the *new* value of the parameter compares to the *old* value of the parameter:

(a) doubled (b) increased by 1 (c) stays the same (d) decreased by 1 (e) halved

Number of entries in the page table: ___e____

Width of each page table entry in bits: ___c____

D) The following program fragment is executed and a record is made of the inputs and outputs of the Memory Management Unit. The record is shown in the table below.

```
sw  x11, 0(x10)
lw  x11, 4(x13)
lui x12, 4
```

Access type	*Virtual address*	*Physical address*
Inst. fetch	`0x60FF8`	`0x10FF8`
Data write	`0x04600`	`0x74600`
Inst. fetch	`0x60FFC`	`0x10FFC`
Data read	`0x18410`	`0x169410`
Inst. fetch	`0x61000`	`0x09000`

Using information from the program and the table above, please deduce the contents of as many entries as possible in the page table. Assume the original address sizes of 64-bit virtual addresses, 40-bit physical addresses, and page size of 4096 (2^{12}) bytes per page. **Assume that pages holding instructions are read-only.**

Please make an entry in the table below for each page table entry we learn about *after* the execution of the program fragment. Note that the table below is not an actual page table, it is just a list of entries from the page table that you can infer from this problem. For each entry provide the VPN, the dirty (D) and resident (R) bits, and the PPN if they are known. If you can't deduce the value of a field, enter a '?' for that field. You may not need to use all the rows of the table below.

VPN	*D*	*R*	*PPN*
0x60	0	1	0x10
0x04	1	1	0x74
0x18	?	1	0x169
0x61	0	1	0x9

E) At some later point in time, suppose that the contents of the page table and its corresponding fully associative TLB are as shown to the right. As previously mentioned, the page table uses an LRU replacement policy, and the LRU page (shown below) will be chosen for replacement if necessary. For each of these four accesses, compute its corresponding physical address and indicate whether the access causes a TLB miss and/or a page fault. **Assume each access below is independent of the others and starts with the TLB and page table state shown to the right.**

TLB

VPN (tag)	V	D	PPN	
0x0	1	0	0x1337	
0x1	0	0	0x7	
0x6	1	1	0x33	**LRU**
0x3	1	0	0x534	

Page Table

VPN	R	D	PPN	
0	1	0	0x1337	
1	0	0	---	
2	1	0	0x450	
3	1	0	0x534	**LRU PAGE**
4	0	0	---	
5	1	1	0xAB5	
6	1	1	0x33	

Fill in table below

	Virt Addr	**PPN (in hex)**	**Phys Addr (in hex)**	**TLB Miss?**	**Page Fault?**
1.	**0x6004**	__0x33____	__0x33004__	Y / [✓ N]	Y / [✓ N]
2.	**0x6030**	__0x33___	__0x33030__	Y / [✓ N]	Y / [✓ N]
3.	**0x1234**	__0x534___	__0x534234_	[✓ Y] / N	[✓ Y] / N
4.	**0x2008**	__0x450___	__0x450008_	[✓ Y] / N	Y / [✓ N]

Problem 8. (OS & Virtual Memory Fa'19 Quiz 3 Problem 1)

(A) A RISC-V system with segmentation-based virtual memory is currently running two processes, A and B, with segment base and bound registers listed below:

Process A: base register = `0x1000` bound register = `0x4000`
Process B: base register = `0x10000` bound register = `0x40000`

The table below lists three virtual addresses. Fill the table by translating the addresses for processes A and B. In each cell, write either the physical address that corresponds to the virtual address, or write SEGFAULT if this virtual address is outside the process's address space (which would cause a segmentation fault, i.e., an out-of-bounds violation).

Hint: Recall that the bound check and the translation from virtual to physical address happen in parallel.

Virtual Address	**Physical address for process A (or SEGFAULT)**	**Physical address for process B (or SEGFAULT)**
`0x500`	0x1500	0x10500
`0x3500`	0x4500	0x13500
`0x10000`	Segfault	0x20000

(B) Each of these instruction sequences experiences an interrupt or an exception that causes the processor to enter privileged mode (i.e., the operating system).

Sequence A

```
  // address 0x0 not resident
  // in virtual memory
  lw x1, 0x0(x0)
```

Sequence B

```
  li x1, 0x1234
  li x2, 0x7
  // this instruction is not in RV32I
  div x3, x1, x2
```

Sequence C

```
  li   x1, 0x10000
  li   x2, 0
  // this loop gets
  // a timer interrupt
loop:
  addi x2, x2, 1
  bne  x1, x2, loop
```

Sequence D

```
endless_loop:
  la a0, string // loads address
  li a7, 0x13   // print_string
syscall number
  ecall
  j endless_loop

string:
  .ascii "Hello from Quiz 3!\n\0"
```

Assume that the OS may only switch processes while servicing timer interrupts. Identify the instruction sequence(s) for which the statement applies. If no sequence applies, indicate NONE.

Which sequences...

(i) enter privileged mode: ___A, B, C, D____

(ii) save x1 through x31 and exception pc for the currently executing process: ____A, B, C, D____

(iii) handle an exception by emulating the instruction at the saved pc of the current process: ____B___________

(iv) handle an exception by loading a page from disk: ____A___________

(v) handle a system call: ____D___________

(vi) add 4 to the saved pc of the current process: ____B, D_________

(vii) subtract 4 from the saved pc of the current process: ____NONE_______

(viii) choose a new current process (which may be the same as the current process): ____C___________

(ix) load registers for the current process, then exit privileged mode and return to the saved pc of the current process: ____A, B, C, D____