Exercises

1) What are three specific benefits of using virtual memory?

Bridges memory and disk in memory hierarchy.
Simulates full address space for each process.
Enforces protection between processes.

2) What should happen to the TLB when a new value is loaded into the page table address register?

The valid bits of the TLB should all be set to 0 (invalidated). The page table entries in the TLB corresponded to the old page table, so none of them are valid once the page table address register points to a different page table.

3) Fill in the following formulas below.

Page offset bits = $\log_2$(page size in bytes)

Virtual address bits = virtual page number bits + page offset bits

Physical address bits = physical page number bits + page offset bits

Virtual page number bits = $\log_2$(number of virtual pages)

Entries in a page table = number of virtual pages

4) Fill in the following table:

VA width (n)	PA width (m)	Page size (P)	VPN width	PPN width	Bits in PTE (assume V, R, W, X)
32	32	16KiB	18	18	22
32	26	8KiB	19	13	17
36	32	32KiB	21	17	21
40	36	32KiB	25	21	25
64	40	64KiB	48	24	28

5) **Processor:** 16-bit addresses, 256 byte pages
TLB: 8-entry fully associative with LRU replacement
- Track LRU using 3 bits to encode the order in which pages were accessed, with 0 being the most recent

At some time instant, the TLB for the current process is in the initial state given below. Assume that all page table entries NOT in the initial TLB start as invalid.
- OS will assign new pages at the lowest available PPN starting at `0x17`

Assume all pages can be read from and written to (ignore protection).

Fill in the final state of the TLB according to the access pattern below:

Access pattern:
1. Read `0x11F0`
2. Write `0x1301`
3. Write `0x20AE`
4. Write `0x2332`
5. Read `0x20FF`
6. Write `0x3415`

Initial TLB:

TLBT	PPN	Valid	LRU	Dirty?
0x01	0x11	1	0	1
0x00	0x00	0	7	0
0x10	0x13	1	1	1
0x20	0x12	1	5	0
0x00	0x00	0	7	0
0x11	0x14	1	4	0
0xAC	0x15	1	2	1
0xFF	0x16	1	3	0

n = 16, p = 8, so VPN width = $n - p$ = 8 bits.
TLB is fully associative, so S = 1 and TLBT = VPN.

1. Read `0x11F0` → TLBT = `0x11`, TLB Hit

TLBT	PPN	Valid	LRU	Dirty?
0x01	0x11	1	1	1
0x00	0x00	0	7	0
0x10	0x13	1	2	1
0x20	0x12	1	5	0
0x00	0x00	0	7	0
0x11	0x14	1	0	0
0xAC	0x15	1	3	1
0xFF	0x16	1	4	0

2. Write `0x1301` → TLBT = `0x13`, TLB Miss
Map `0x13` to PPN `0x17` & fill invalid slot

TLBT	PPN	Valid	LRU	Dirty?
0x01	0x11	1	2	1
0x13	0x17	1	0	1
0x10	0x13	1	3	1
0x20	0x12	1	6	0
0x00	0x00	0	7	0
0x11	0x14	1	1	0
0xAC	0x15	1	4	1
0xFF	0x16	1	5	0

3. Write 0x20AE → TLBT = 0x20, TLB Hit

TLBT	PPN	Valid	LRU	Dirty?
0x01	0x11	1	3	1
0x13	0x17	1	1	1
0x10	0x13	1	4	1
0x20	0x12	1	0	1
0x00	0x00	0	7	0
0x11	0x14	1	2	0
0xAC	0x15	1	5	1
0xFF	0x16	1	6	0

4. Write 0x2332 → TLBT = 0x23, TLB Miss
Map 0x23 to PPN 0x18 & fill invalid slot

TLBT	PPN	Valid	LRU	Dirty?
0x01	0x11	1	4	1
0x13	0x17	1	2	1
0x10	0x13	1	5	1
0x20	0x12	1	1	1
0x23	0x18	1	0	1
0x11	0x14	1	3	0
0xAC	0x15	1	6	1
0xFF	0x16	1	7	0

5. Read 0x20FF → TLBT = 0x20, TLB Hit

TLBT	PPN	Valid	LRU	Dirty?
0x01	0x11	1	4	1
0x13	0x17	1	2	1
0x10	0x13	1	5	1
0x20	0x12	1	0	1
0x23	0x18	1	1	1
0x11	0x14	1	3	0
0xAC	0x15	1	6	1
0xFF	0x16	1	7	0

6. Write 0x3415 → TLBT = 0x34, TLB Miss
Map 0x34 to PPN 0x19 and replace LRU

TLBT	PPN	Valid	LRU	Dirty?
0x01	0x11	1	5	1
0x13	0x17	1	3	1
0x10	0x13	1	6	1
0x20	0x12	1	1	1
0x23	0x18	1	2	1
0x11	0x14	1	4	0
0xAC	0x15	1	7	1
0x34	0x19	1	0	1

Final TLB:

TLBT	PPN	Valid	LRU	Dirty?
0x01	0x11	1	5	1
0x13	0x17	1	3	1
0x10	0x13	1	6	1
0x20	0x12	1	1	1
0x23	0x18	1	2	1
0x11	0x14	1	4	0
0xAC	0x15	1	7	1
0x34	0x19	1	0	1