DEPARTMENT OF
NETWORKED SYSTEMS
AND SERVICES

COMPUTER ARCHITECTURES

Practical Tasks in:
Memory Technology and Virtual Memory

Gábor Lencse

BUTE Department of Networked Systems and Services
lencse@hit.bme.hu

Budapest,
2024. 04. 11.

MŰEGYETEM 1782

Memory Technology

TASK 1

- Assume that the memory of a computer is based on single channel DDR1-SDRAM technology with 64 bit data units. The burst size is set to 8. Let us ignore the presence of multiple banks and ranks.
- The timing values of the memory modules are as follows (given in clocks):
  - T_{RP} = 4 (the delay of the PRECHARGE command)
  - T_{RCD} = 5 (the time needed to activate a row)
  - T_{CAS} = 9 (the time needed to read/write a column in the active row)
- The memory controller receives the following read requests (of 64 bytes, hence one burst), given by row and column coordinates:
  - (row 3, column 8), (row 3, column 2), (row 7, column 9)

TASK 1

- Row 7 is active initially. After the last command, the memory controller does not close the active row.

a) List the commands sent by the memory controller to the memory modules (in the right order) according to FCFS and FR-FCFS command scheduling!

b) According to the FCFS scheduling, in which clock cycle does the first data appear on the data bus? And when does the last data corresponding to request (row 3, column 2) appear?

TASK 1/A/FCFS – SOLUTION

- The original read requests (row 7 is open):

  (row 3, column 8)
  (row 3, column 2)
  (row 7, column 9)

- Commands of FCFS scheduling:

  PRECHARGE
  ACTIVATE 3
  READ 8
  READ 2
  PRECHARGE
  ACTIVATE 7
  READ 9

TASK 1/A/FR-FCFS – SOLUTION

- The original read requests (row 7 is open):

  (row 3, column 8)
  (row 3, column 2)
  (row 7, column 9)

- Let us reorder the read commands:

  (row 7, column 9)
  (row 3, column 8)
  (row 3, column 2)

- Commands of FR-FCFS scheduling:

  READ 9
  PRECHARGE
  ACTIVATE 3
  READ 8
  READ 2

TASK 1/B/(3,8 FIRST) – SOLUTION

- According to the FCFS scheduling, in which clock cycle does the first data appear on the data bus?
- Let us sum up the time of the following commands:

  PRECHARGE: $T_{RP} = 4$
  ACTIVATE 3: $T_{RCD} = 5$
  READ 8: $T_{CAS} = 9$

  4+5+9=18 clock cycles

TASK 1/B/(3,2 LAST) – SOLUTION

- And when does the last data corresponding to request (row 3, column 2) appear?
- As we have seen, 18 clock cycles elapsed until the the first data appeared.
- Burst length is 8, DDR means that there is data transmission both at the rising edge and at the falling edge of the clock signal.
- Therefore, one burst last for 8/2=4 clock cycles
- And we consider two bursts: (3, 8) and (3, 2)
- In all: 18+4+4=26 clock cycles

Virtual Memory

TASK 2

- Let the virtual addresses be 16 bit wide, and the physical addresses 15 bit wide. The page size is 2^{12} byte = 4 kB, and the page table is a simple single-level page table with 8 bit wide entries.

a) In the virtual address, how many bits correspond to the page number and the position inside the page (offset)?

b) What is the total size of the page table?

c) How many pages fit into the physical memory?

TASK 2

d) Assume the current state of the page table is as follows. Where are pages 3, 6, and 11 located?

e) Modify the content of the page table according to the following.
- Page 6 is on the disk
- Page 11 is in the physical memory, in frame 1
- Page 8 is on the disk
- Page 2 is in the physical memory, in frame 6

f) If there is no page fault, how many memory operations are required for address translation
- in case of TLB hit?
- in case of TLB miss?

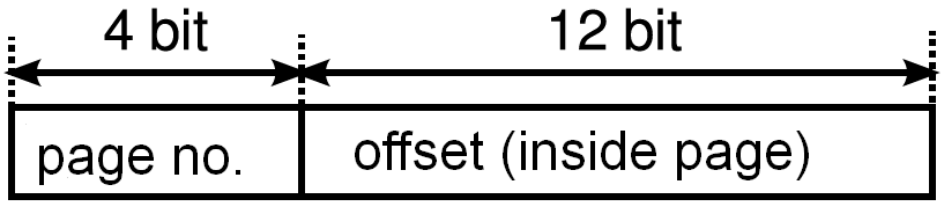
Page table:

	Valid:	Frame:
0:	1	5
1:	1	7
2:	0	?
3:	1	2
4:	0	?
5:	0	?
6:	1	1
7:	0	?
8:	1	6
9:	0	?
10:	0	?
11:	0	?
12:	1	0
13:	1	3
14:	0	?
15:	1	4

TASK 2 – SOLUTION

- Let the virtual addresses be 16 bit wide, and the physical addresses 15 bit wide. The page size is 2^{12} byte = 4 kB, and the page table is a simple single-level page table with 8 bit wide entries.

a) In the virtual address, how many bits correspond to the page number and the position inside the page (offset)?



b) What is the total size of the page table?

2^4=16 entries, 1byte/entry, that is: 16 bytes

c) How many pages fit into the physical memory?

Phys. mem. size: 2^{15}=32kB, 32kB/4kB=8 pages

TASK 2 – SOLUTION

d) Assume the current state of the page table is as follows. Where are pages 3, 6, and 11 located?

- We need to check the right entries:
  - Page 3: Valid bit is 1, it is in fame 2
  - Page 6: Valid bit is 1, it is in frame 1
  - Page 11: Valid bit is 0, it is on the disk

Page table:

	Valid:	Frame:
0:	1	5
1:	1	7
2:	0	?
3:	1	2
4:	0	?
5:	0	?
6:	1	1
7:	0	?
8:	1	6
9:	0	?
10:	0	?
11:	0	?
12:	1	0
13:	1	3
14:	0	?
15:	1	4

TASK 2 – SOLUTION

e) Modify the content of the page table according to the following.

- Page 6 is on the disk
- Page 11 is in the physical memory, in frame 1
- Page 8 is on the disk
- Page 2 is in the physical memory, in frame 6

Page table:

	Valid:	Frame:
0:	1	5
1:	1	7
2:	~~0~~ 1	~~?~~ 6
3:	1	2
4:	0	?
5:	0	?
6:	~~1~~ 0	~~1~~ ?
7:	0	?
8:	~~1~~ 0	~~6~~ ?
9:	0	?
10:	0	?
11:	~~0~~ 1	~~?~~ 1
12:	1	0
13:	1	3
14:	0	?
15:	1	4

TASK 2 – SOLUTION

f) If there is no page fault, how many memory operations are required for address translation
   - in case of TLB hit?
     - 0 (We do not need to touch the page table at all.)
   - in case of TLB miss?
     - 1 (This is the advantage of the single-level page table.)

TASK 3

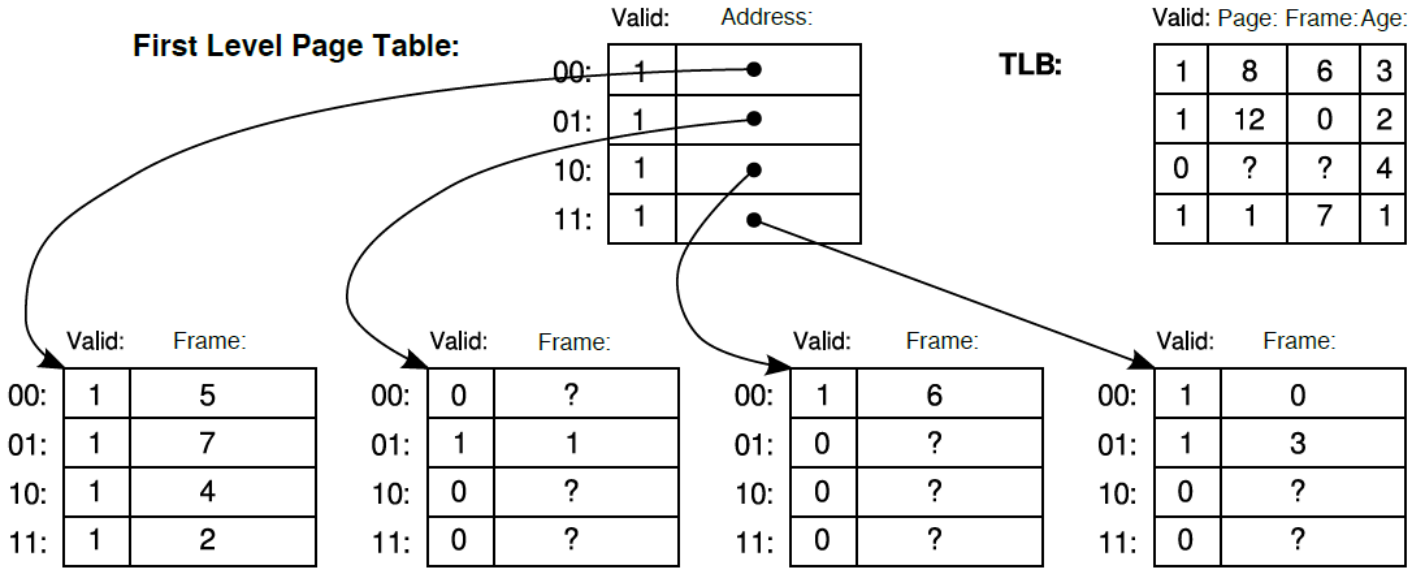
- Let the virtual addresses be 14-bit wide, and the physical addresses 13-bit wide. The page size is 2^{10} byte = 1 kB, and the page table is a two-level hierarchical page table with 16-bit wide entries. To accelerate the address translation, the CPU has a TLB with 4 entries and LRU (Least Recently Used) content management.
  a) In the virtual address, how many bits correspond to the page number, the page table indices, and the position inside the page (offset)?
  b) How many pages fit into the physical memory?
  c) What is the total size of the page table?
  d) What is the minimal size of the page table? How much less is it compared to a single-level page table?

TASK 3

e) If there is no page fault, how many memory operations are required for address translation
   - in case of TLB hit?
   - in case of TLB miss?

f) Assume that the current program executed by the CPU refers to the following pages (in this order):
   - 13, 12, 7, 2, 13.

The initial state of the page table and the TLB is the following:



TASK 3

- Follow the evolution of the page table and the TLB with the given memory references! When the operating system needs to accommodate a new page in the physical memory, it has to throw out another page to make place. The next victims are pages 1 and 5. After modifying the page table, the operating system invalidates the entire TLB.
- What is the number of TLB hits?
- What is the number of page faults?

TASK 3 – SOLUTION

- Let the virtual addresses be 14-bit wide, and the physical addresses 13-bit wide. The page size is 2^{10} byte = 1 kB, and the page table is a two-level hierarchical page table with 16-bit wide entries. To accelerate the address translation, the CPU has a TLB with 4 entries and LRU content management.
  a) In the virtual address, how many bits correspond to the page number, the page table indices, and the position inside the page (offset)?
     - Offset: 10 bits
     - Page number: 14-10 = 4 bits
     - Page table indices: 4/2 = 2 bits

TASK 3 – SOLUTION

b) How many pages fit into the physical memory?
- The size of the physical memory is: 2^{13} = 8kB
- The size of the pages 2^{10} = 1kB
- 8kB/1kB = **8 pages**

c) What is the total size of the page table?
- Level 1 page table: 4 entries, 2bytes / entry: 4*2 = 8bytes
- Level 2 page tables: there are 4 of them, 8bytes each, 4*8 = 32 bytes
- Total size: 8+32 = **40 bytes**

d) What is the minimal size of the page table?
- Optimal case: we have 1 page table at each level: 2*8 = **16 bytes**

e) How much less is it compared to a single-level page table?
- 2^4 = 16 entries, 16*2bytes = **32 bytes**
- 16 / 32 = ½ – it is only the **half of it**.

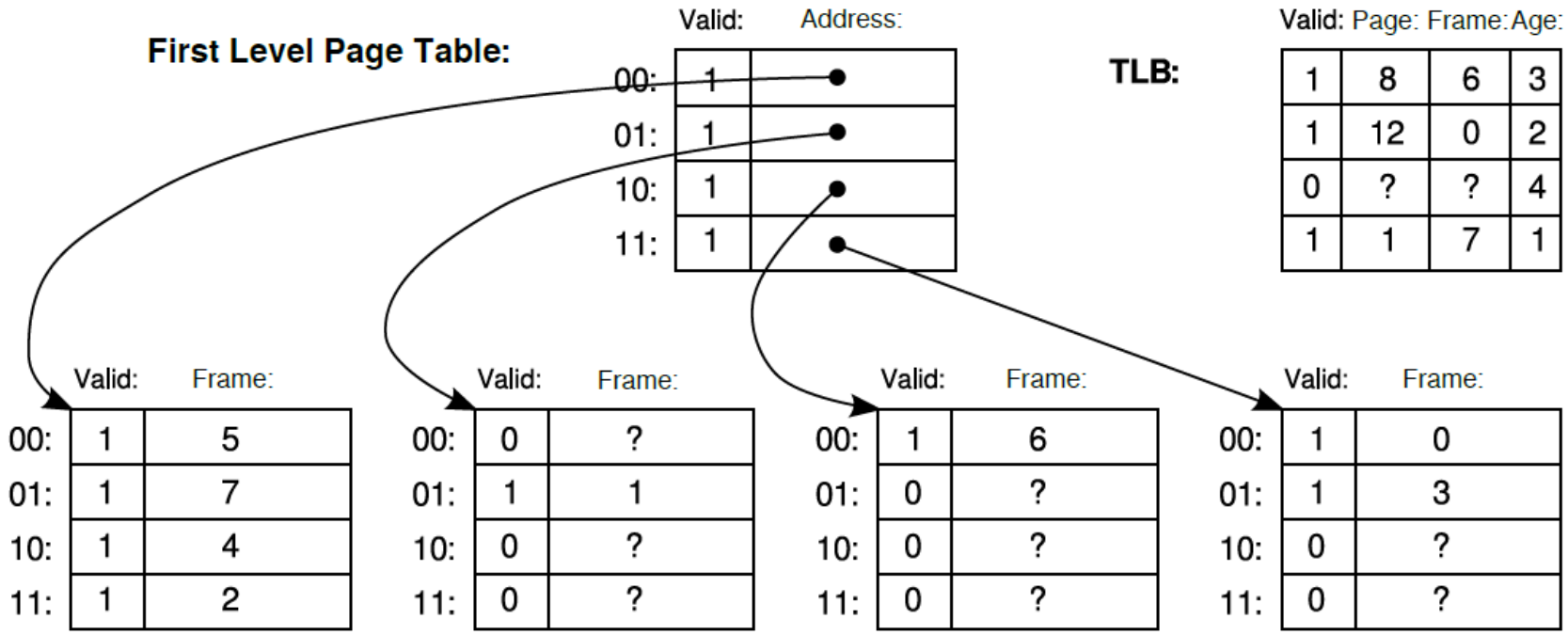
TASK 3 – SOLUTION

e) If there is no page fault, how many memory operations are required for address translation
- in case of TLB hit?
  - 0 (We do not need to touch the page table at all.)
- in case of TLB miss?
  - 2 (One at each level of the page table.)

TASK 3 – SOLUTION

f) Assume that the current program executed by the CPU refers to the following pages (in this order):
   - 13, 12, 7, 2, 13.

The initial state of the page table and the TLB is the following:



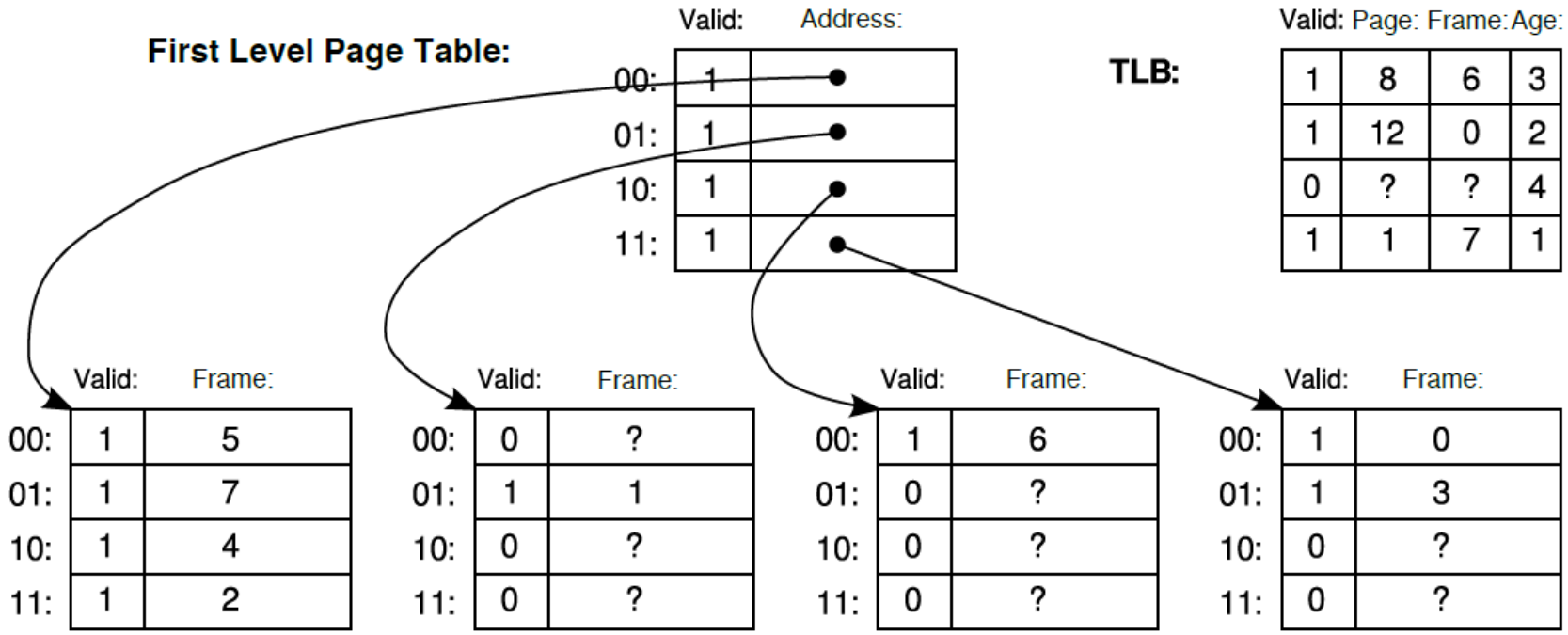
TASK 3 – SOLUTION

f) Assume that the current program executed by the CPU refers to the following pages (in this order):
- 13, 12, 7, 2, 13.

Let us write the page numbers in binary format:
- 13: 1101, that is level 1 index: 11, level 2 index: 01
- 12: 1100, that is level 1 index: 11, level 2 index: 00
- 7: 0111, that is level 1 index: 01, level 2 index: 11
- 2: 0010, that is level 1 index: 00, level 2 index: 10

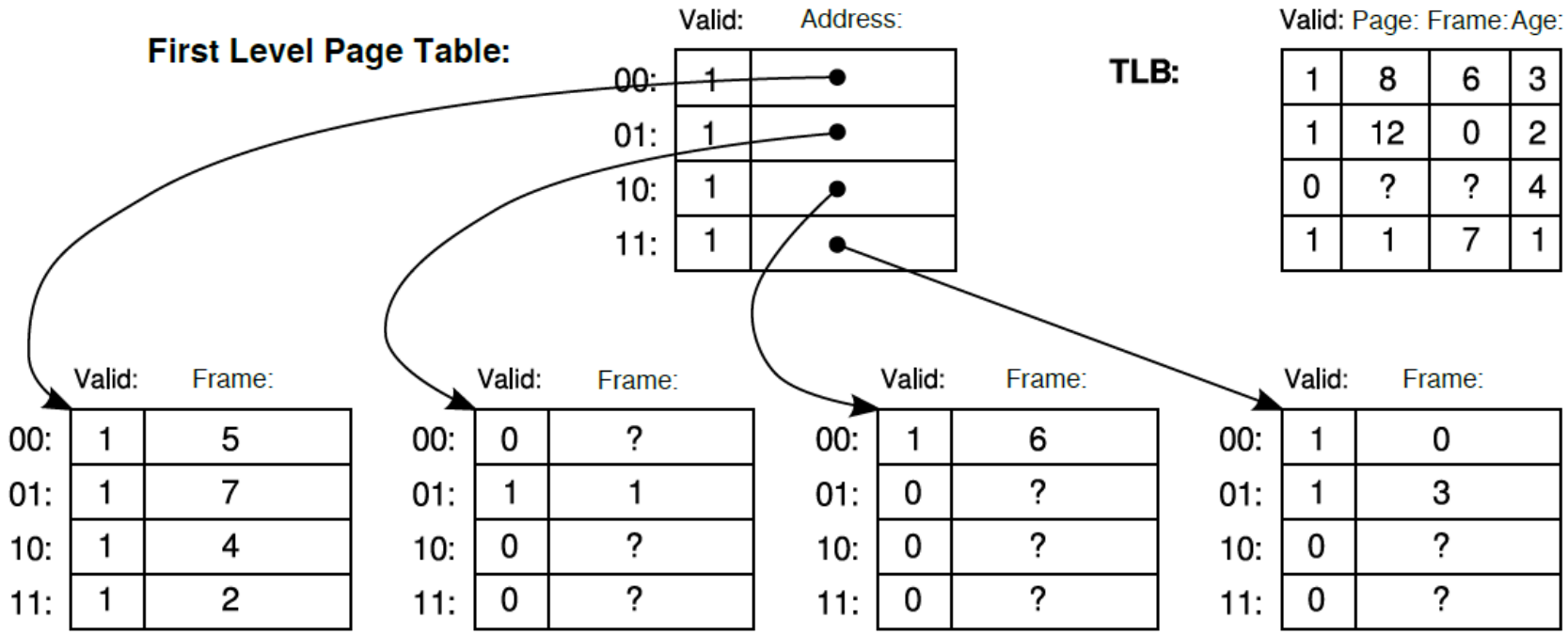
TASK 3 – SOLUTION

Page number **13** is NOT in the TLB. There is a TLB miss. Let us follow: FLPT entry 11, SLPT entry 01 is **valid**, and shows that page 13 is in frame **3**



TASK 3 – SOLUTION

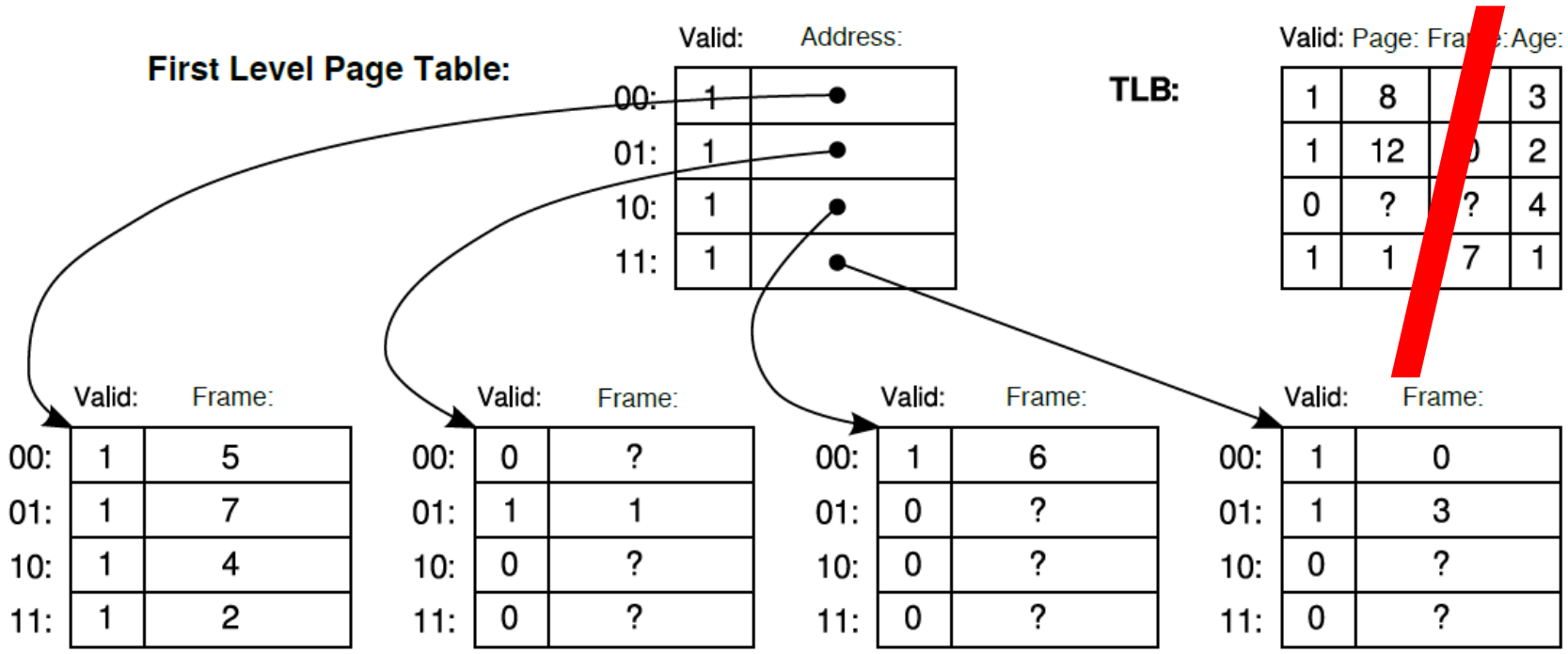
Page number **13** is NOT in the TLB. There is a TLB miss.
Let us follow: FLPT entry 11, SLPT entry 01 is **valid**, and shows that page 13 is in frame **3**
We must put 13 ↔ 3 into the TLB, it will be:
(all other entries are to be aged)



TASK 3 – SOLUTION

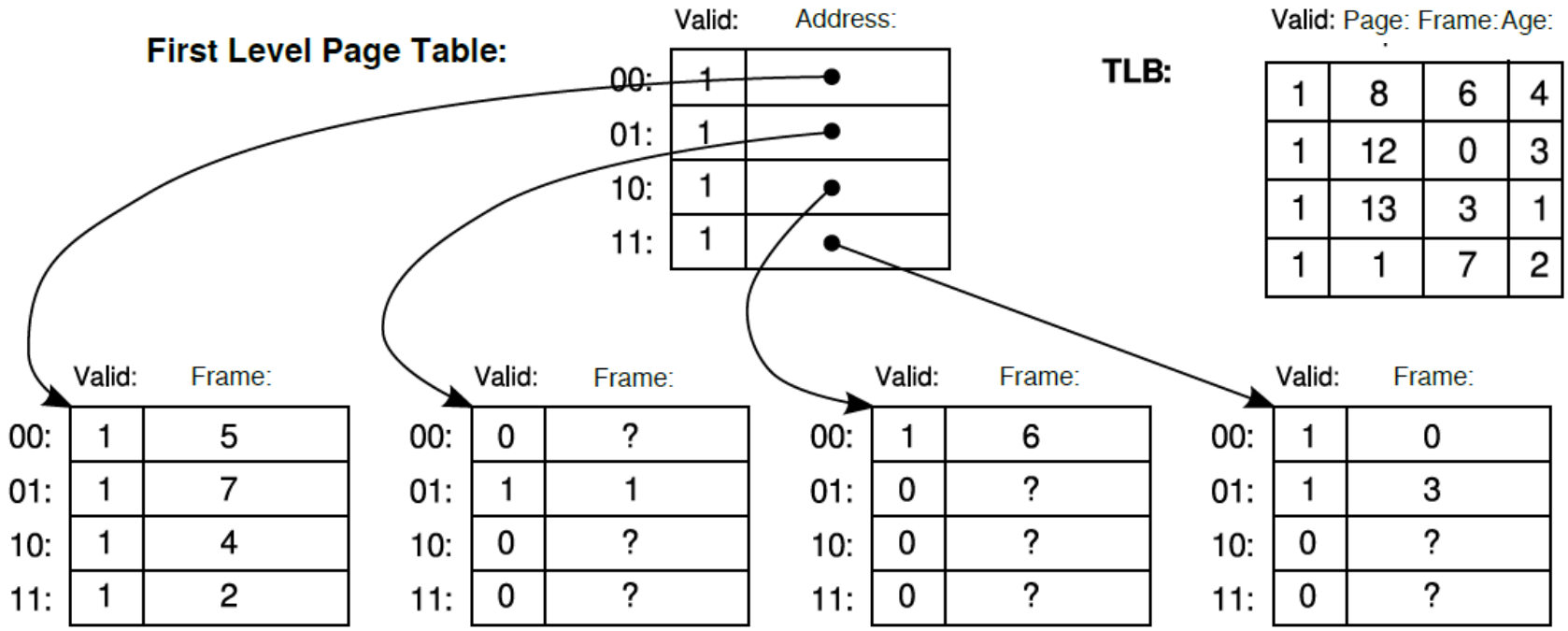
Page number **13** is NOT in the TLB. There is a TLB miss.
Let us follow: FLPT entry 11, SLPT entry 01 is **valid**, and shows that page 13 is in frame **3**
We must put 13 ↔ 3 into the TLB, it will be:
(all other entries were aged)

1	8	6	4
1	12	0	3
1	13	3	1
1	1	7	2



TASK 3 – SOLUTION

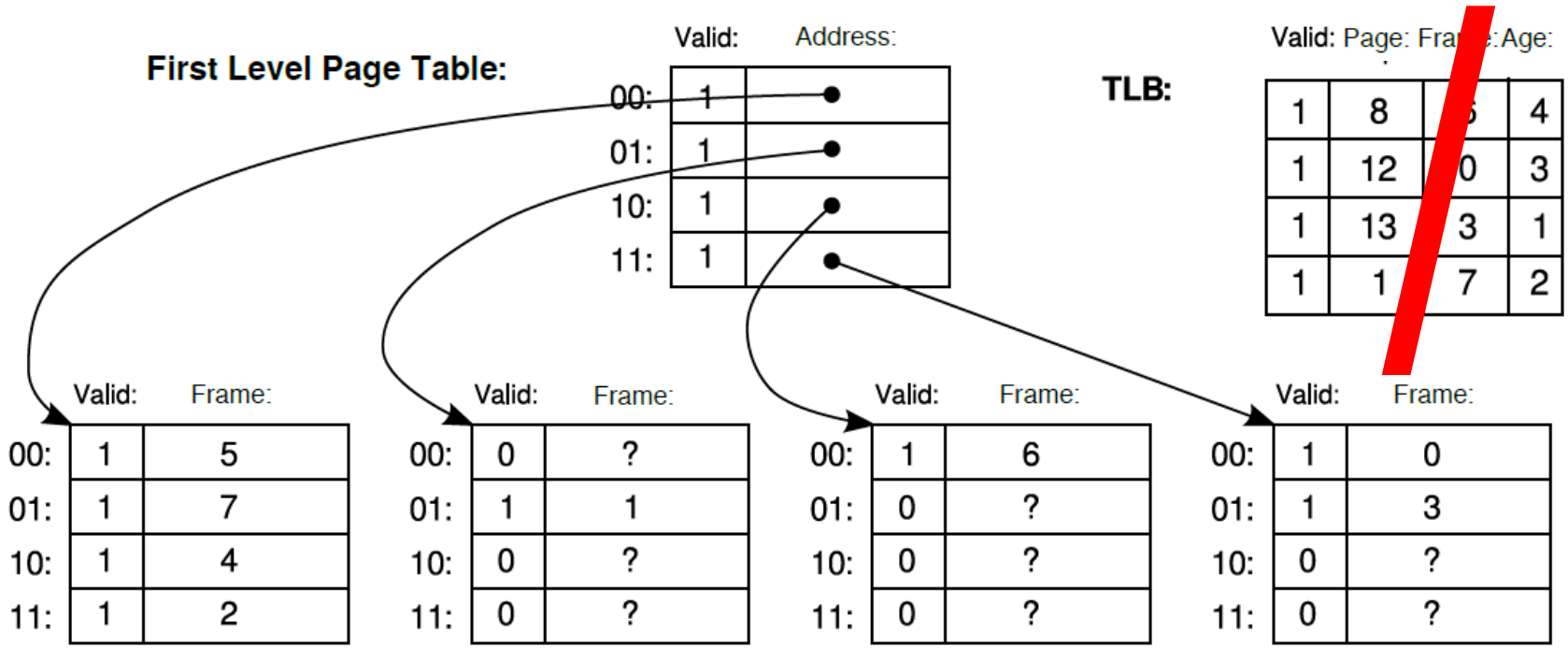
Page number **12** is in the TLB. There is a TLB hit.
The found entry will be the most recently used one (age=1):
(all other entries are to be aged)



TASK 3 – SOLUTION

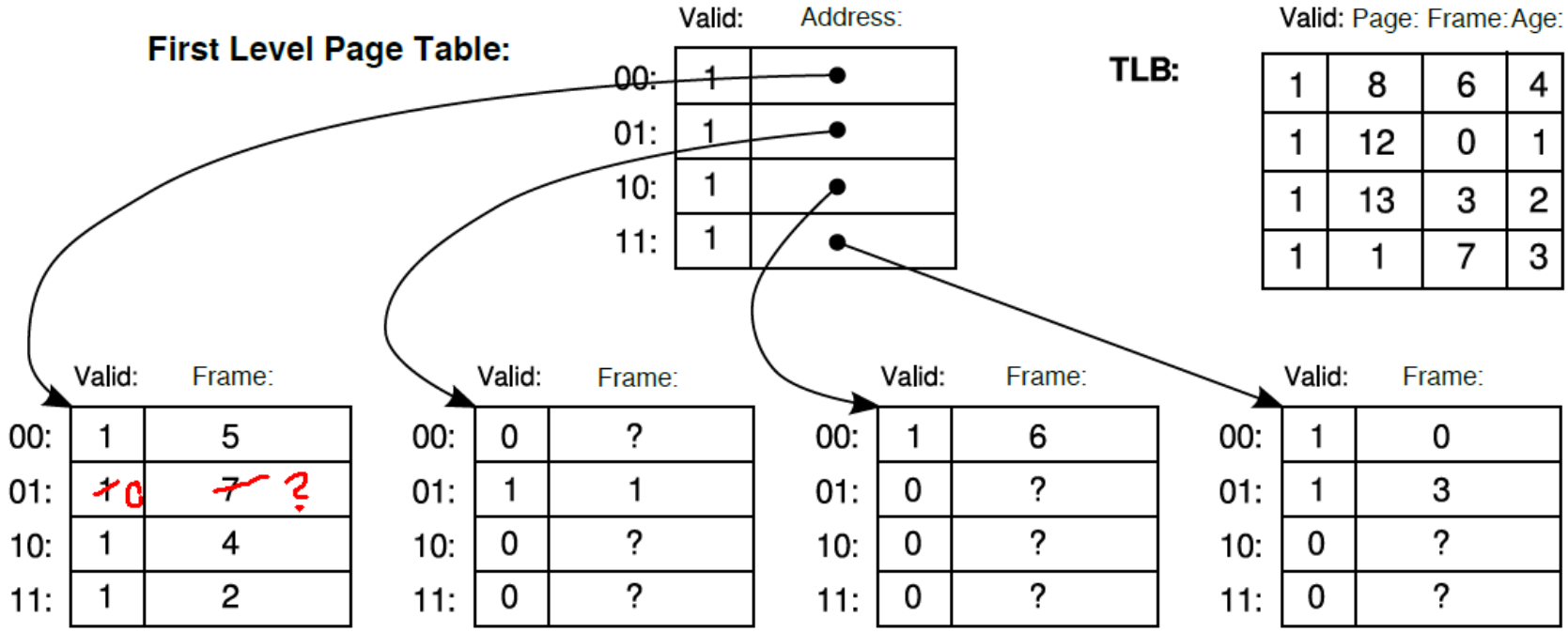
Page number **12** is in the TLB. There is a TLB hit.
The found entry will be the most recently used one (age=1):
(all other entries were aged)

1	8	6	4
1	12	0	1
1	13	3	2
1	1	7	3



TASK 3 – SOLUTION

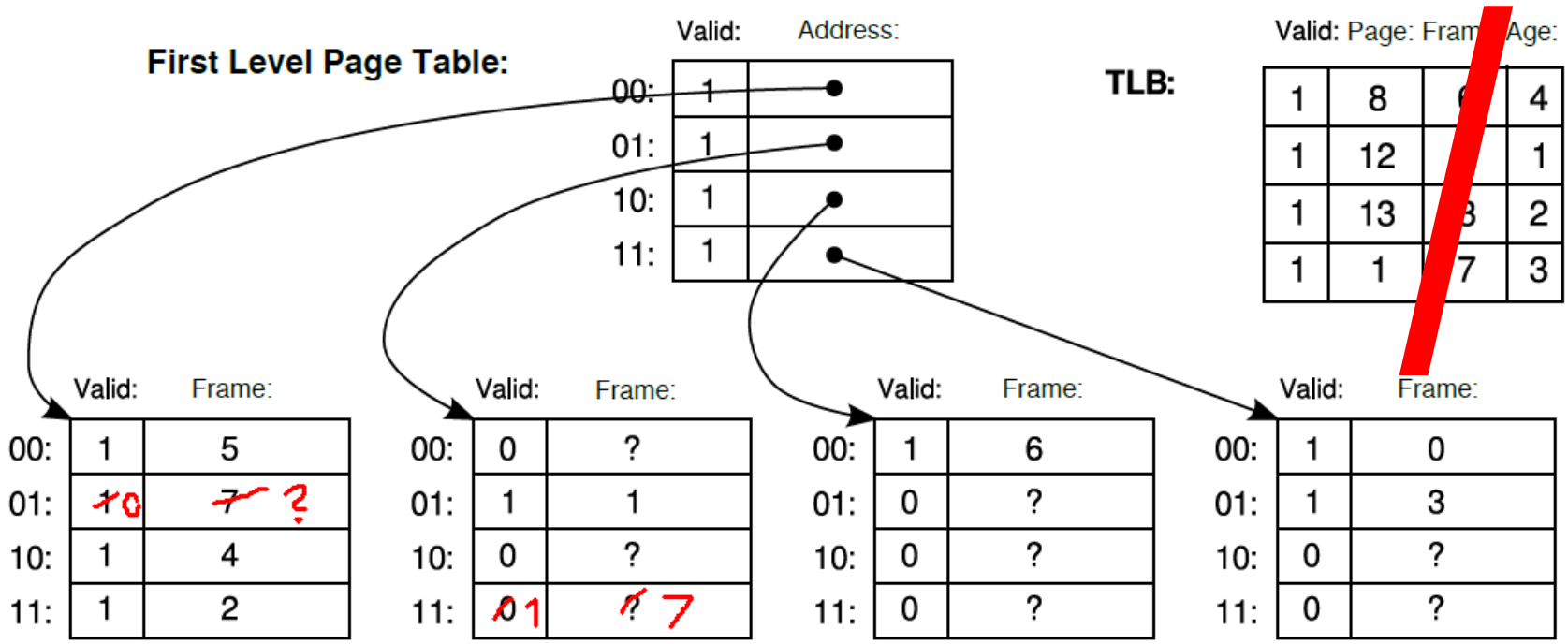
Page number **7** is NOT in the TLB. There is a TLB miss.
Let us follow: FLPT entry 01, SLPT entry 11 is in **invalid**!
There is a **page fault**. We must bring it into the memory.
To be able to do so, we must first delete a page from the memory. According to the task, the first victim will be page 1 (1: 0001) Let us follow FLPT 00, SLPT 01 is in frame **7**
Now we invalidate this entry, and thus make frame 7 free.



TASK 3 – SOLUTION

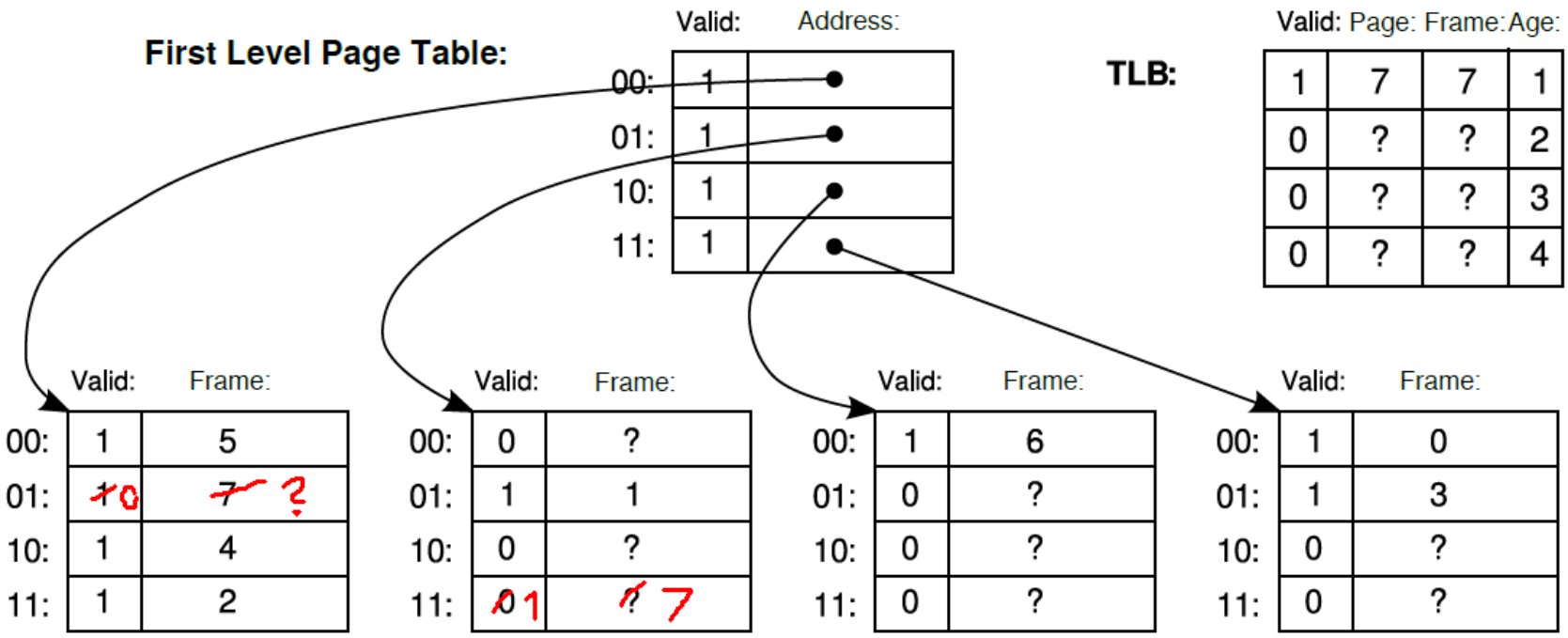
Now, we can bring in page number 7 into the currently freed frame 7. We register its location into the page table.
We also need to invalidate TLB and then register 7 ↔ 7 into the TLB as follows:

1	7	7	1
0	?	?	2
0	?	?	3
0	?	?	4



TASK 3 – SOLUTION

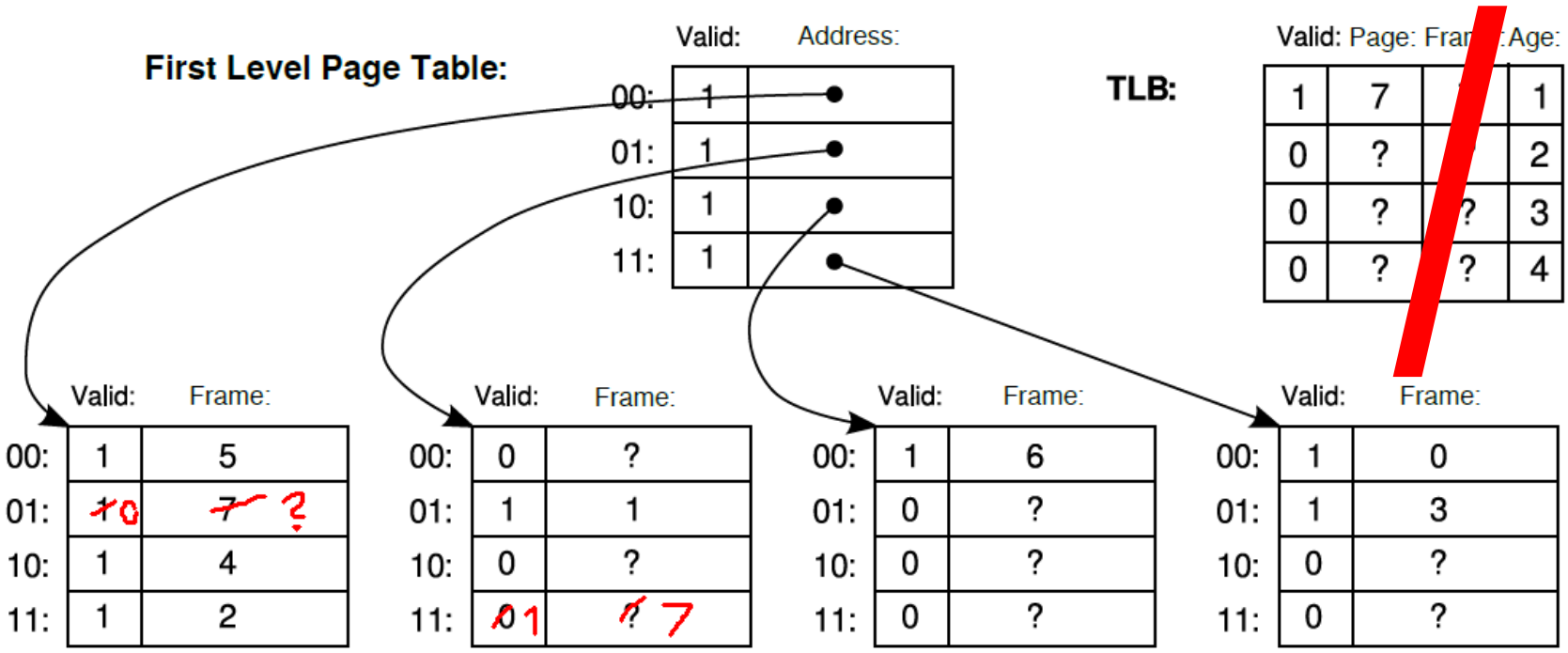
Page number **2** is NOT in the TLB. There is a TLB miss.
Let us follow: FLPT entry 00, SLPT entry 10 is valid, and it shows that page 2 is in frame **4**.
We must put 2 ↔ 4 into the TLB, it will be:
(all valid entries are to be aged)



TASK 3 – SOLUTION

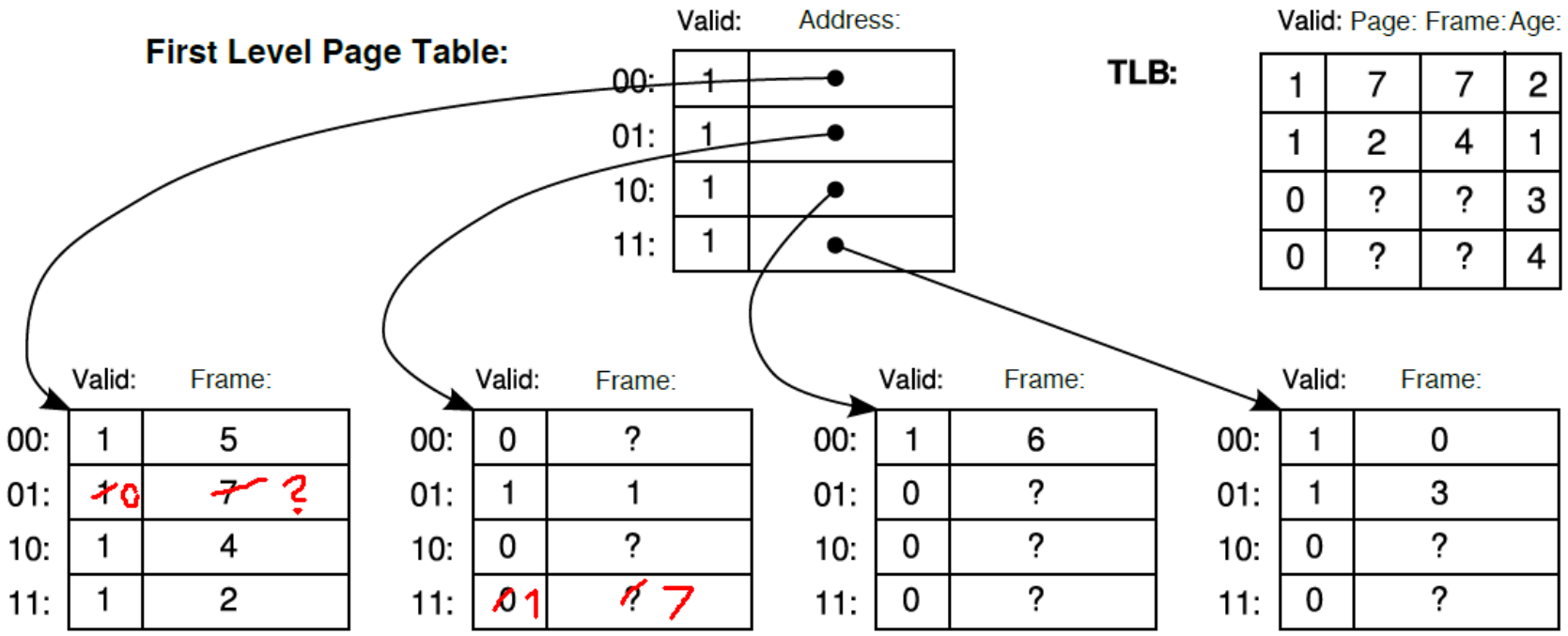
Page number **2** is NOT in the TLB. There is a TLB miss.
Let us follow: FLPT entry 00, SLPT entry 10 is valid, and it shows that page 2 is in frame **4**.
We must put 2 ↔ 4 into the TLB, it will be:
(all valid entries were aged)

1	7	7	2
1	2	4	1
0	?	?	3
0	?	?	4



TASK 3 – SOLUTION

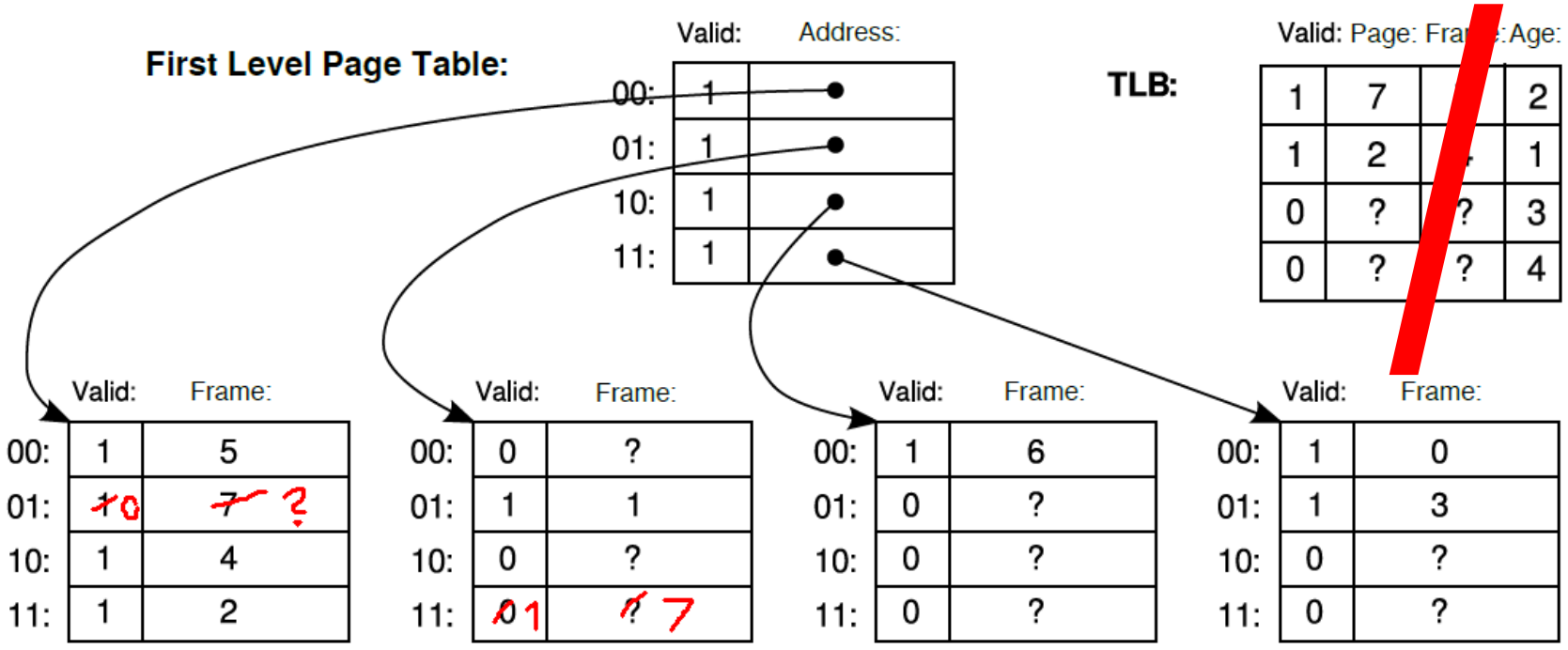
Page number **13** is NOT in the TLB. There is a TLB miss.
Let us follow: FLPT entry 11, SLPT entry 01 is valid, and it shows that page 13 is in frame **3**.
We must put 13 ↔ 3 into the TLB, it will be:
(all valid entries are to be aged)



TASK 3 – SOLUTION

Page number **13** is NOT in the TLB. There is a TLB miss.
Let us follow: FLPT entry 11, SLPT entry 01 is valid, and it shows that page 13 is in frame **3**.
We must put 13 ↔ 3 into the TLB, it will be:
(all valid entries were aged)

1	7	7	3
1	2	4	2
1	13	3	1
0	?	?	4



TASK 3 – SOLUTION

- What is the number of TLB hits?
  - There was a single TLB hit.
- What is the number of page faults?
  - There was a single page fault.