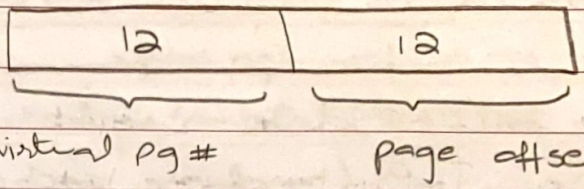
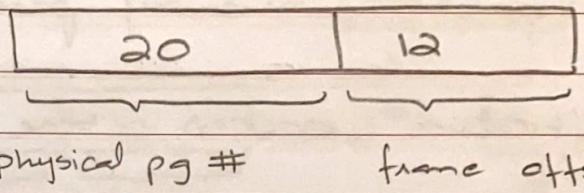
day/date

Q1) page size : 4KB = $2^2 \cdot 2^{10}$ = 12 bits

24 bits : virtual addr.



32 bits : Physical addr.



i) 0010 1011 1100 0101

2b offset = c5

2b = fc , using Page table

⇒ 0x fcc5

ii) 0001 0011 0001 1100

1c

13 = a3 ⇒ 0x a3 1c

iii) 1110 0100 1010 0011

e4 = e5 a3 ⇒ 0x e5a3

iv) 0010 1011 0001 0111

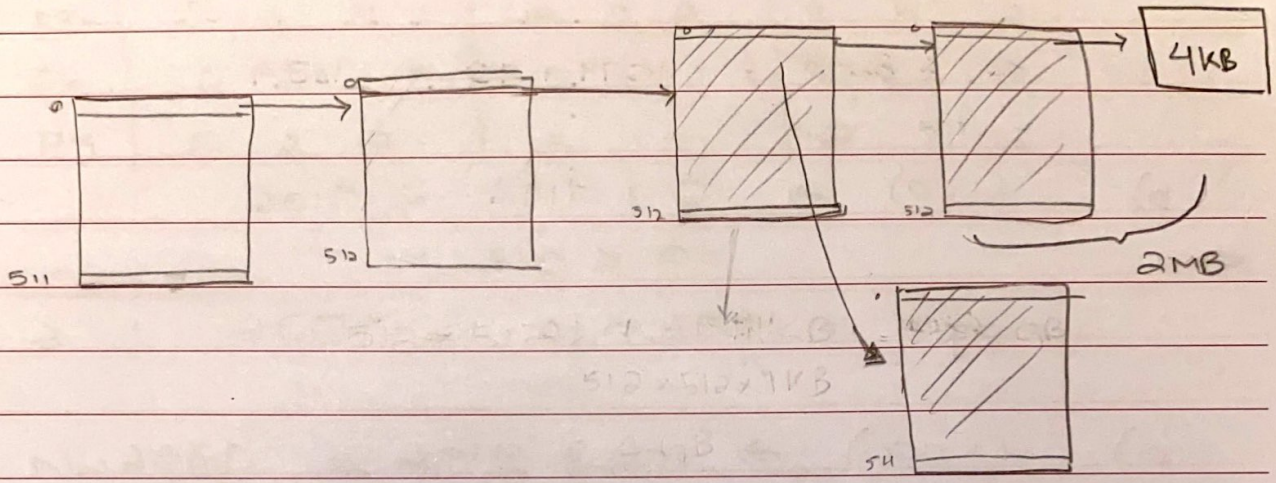
2b = fc 17 ⇒ 0x fc17

day/date

Q2) Virtual address = 48 bits

address division = 9, 9, 9, 9, 12

- page size = $2^2 \cdot 2^{10}$ = 4KB = 4096 bytes
- entries on outer, inner tables = 2^9 = 512 entries



⇒ 32MB / 2MB = 16 pages

3 + 16 = 19 page tables

outer table inner tables

2^9 x 8 bytes = 4KB

⇒ 19 x 4KB = **77 KB**

➡ 16 fourth level tables and 1 first, second, third level tables are required for a process using 32 MB

day/date

(Q3) Parent process executing first:

signal(Y), 60, wait(X) { signal(X), 40, 30, wait(Z), 45

signal(X), 40, 10, 80, Z

⇒ 60, 40, 30, 10, 80, 45
60, 40, 10, 80, 30, 45
60, 10, 80, 40, 30, 45
60, 10, 40, 30, 80, 45

Child process executing first:

signal(X), 40, wait(Y) { signal(Y), 30, wait(Z) } 45

", 60, wait(X), 10, 80 } signalZ

⇒ 40, 30, 60, 10, 80, 45
40, 60, 10, 80, 30, 45
40, 60, 30, 10, 80, 45
40, 60, 10, 30, 80, 45

$\log_2 64 = 6$

day/date

Q6)

10	6
virtual pg #	pg offset

logical address

a) $(\underset{s}{0}, \underset{d}{50})$

$\Rightarrow$ offset + base = 50 + 1024 = 1074

0000 0100 00	11 0010
pg #	offset

virtual pg number = 0100 00 = 16

physical pg number = 16 + 10 = 26 (11010)

$\Rightarrow$

0000 0110.10	11 0010
frame #	frame offset

day/date

b) (1, 0)

⇒ 4196 + 0 = 4196

0001 0000 0110 0100
pg# offset

virtual pg #: 65
physical frame #: 75

⇒ 1001 0111 0100
frame # offset

c) (1, 100)

⇒ 4196 + 100 = 4296

0001 0000 1100 1000
pg # offset

67 + 10 = 77

⇒ 0001 0011 0100 1000

d) (1, 700)

error; since 700 is greater than the length of segment 1, which is 512.

day/date

s , d

f) (3, 200)

2048 + 200 = 2248

0000 1000 1100 1000

= 35 + 10 offset

= 45

⇒ 0000 1011 0100 1000

frame # frame offset

e) (2, 10)

10 + 128 = 138

~~138~~ ~~69~~ ~~34~~ ~~17~~ ~~8~~ ~~4~~ ~~2~~ 1

~~2~~

00 0000 0010 1010

offset

pg = 2 + 10 = 12

⇒ 00 0000 1100 00 1010

frame # offset

day/date

Existing:

Q7) A (9), B (6), C (6)

	max			alloc			need		
P1	5	4	2	1	2	1	4	2	1
P2	3	1	3	1	0	2	2	1	1
P3	6	6	4	2	2	2	4	4	2
P4	2	2	1	0	0	1	2	2	0
P5	5	2	2	3	1	0	2	1	2

available

2 1 0

⇒ Not safe as no process has a row in need matrix that is less than or equal to available, hence it is unsafe.

day/date

(Q8) existing: A B C
10 6 4

	alloc	request	available
			3 3 0
~~P1~~	~~1 0 0~~	~~0 0 1~~	4 3 1
P2	3 0 1	0 4 0	5 3 1
P3	0 2 2	3 1 2	
~~P4~~	~~1 0 1~~	~~2 3 0~~	
P5	2 1 0	3 3 3	

P4 , P1

after this, no row can be satisfied

thus; P2, P3, P5 are deadlocked processes

day/date

Q 9

```
monitor SemMonitor {
    int count;
    condition c;

    // signal (S)
    public void S() {
        count++;
        c.signal()
    }

    // wait(S)
    public void W() {
        if (count == 0)
            c.wait();
        count--;
    }
```