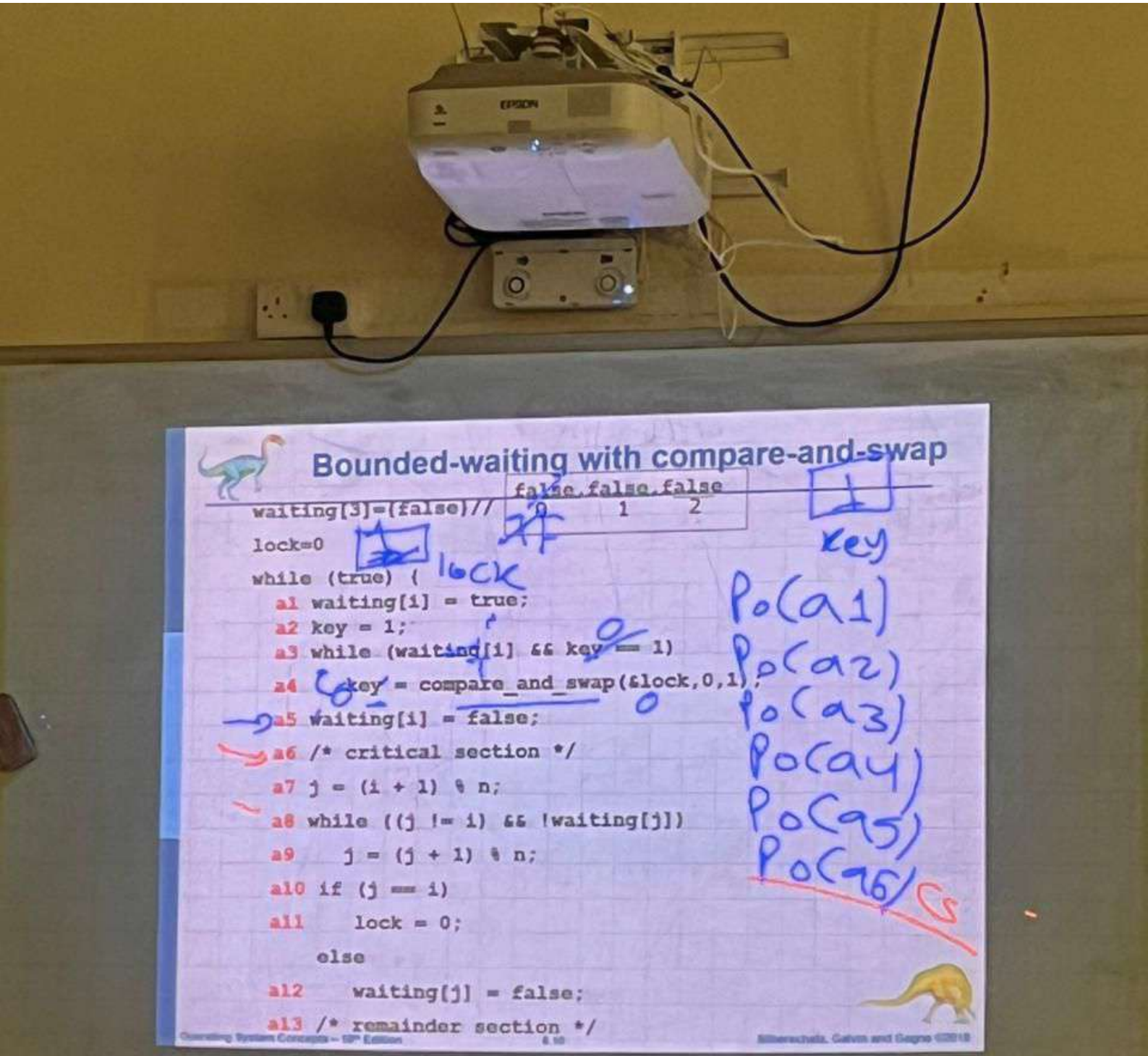
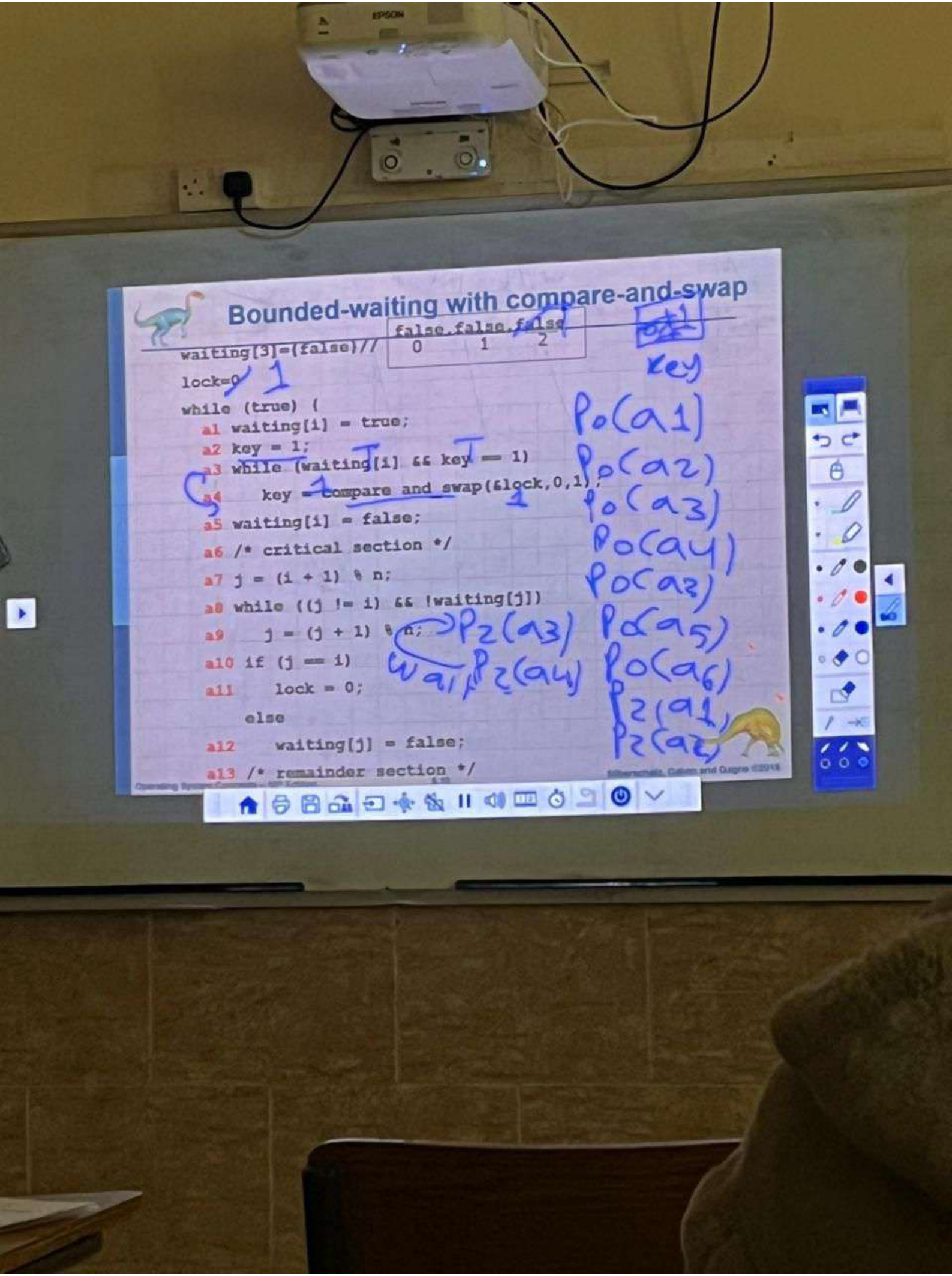
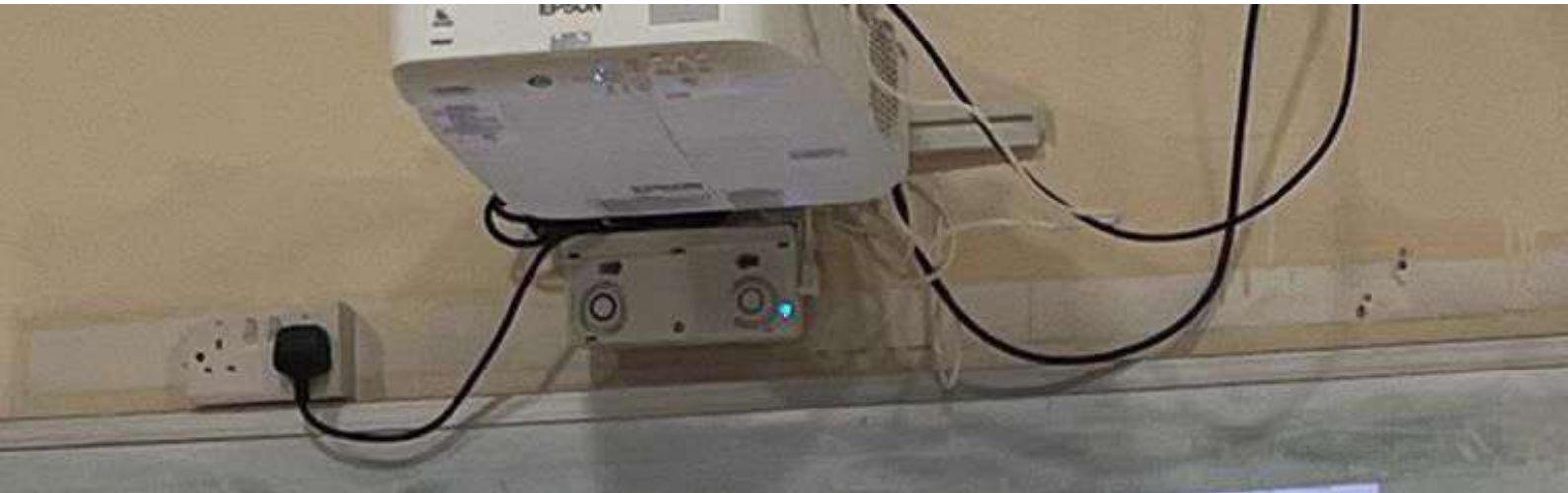

Bounded-waiting with compare-and-swap
false false false
0 1 2
waiting[j]=(false)//
lock=0
while (true) {
a1 waiting[i] = true;
a2 key = 1;
a3 while (waiting[i] && key == 1)
a4 key = compare_and_swap(&lock,0,1);
a5 waiting[i] = false;
a6 /* critical section */
a7 j = (i + 1) % n;
a8 while ((j != i) && !waiting[j])
a9 j = (j + 1) % n;
a10 if (j == i)
a11 lock = 0;
else
a12 waiting[j] = false;
a13 /* remainder section */
lock
key
Po(a1)
Po(a2)
Po(a3)
Po(a4)
Po(a5)
Po(a6)
CS


Bounded-waiting with compare-and-swap
waiting[3]=(false)//
false false false
0 1 2
lock=0
while (true) {
a1 waiting[i] = true;
a2 key = 1;
a3 while (waiting[i] && key == 1)
a4 key = compare and swap(&lock,0,1);
a5 waiting[i] = false;
a6 /* critical section */
a7 j = (i + 1) % n;
a8 while ((j != i) && !waiting[j])
a9 j = (j + 1) % n;
a10 if (j == i)
a11 lock = 0;
else
a12 waiting[j] = false;
a13 /* remainder section */
key
P0(a1)
P0(a2)
P0(a3)
P0(a4)
P0(a3)
P0(a5)
P0(a6)
P2(a1)
P2(a2)
P2(a3)
Wait
P2(a4)

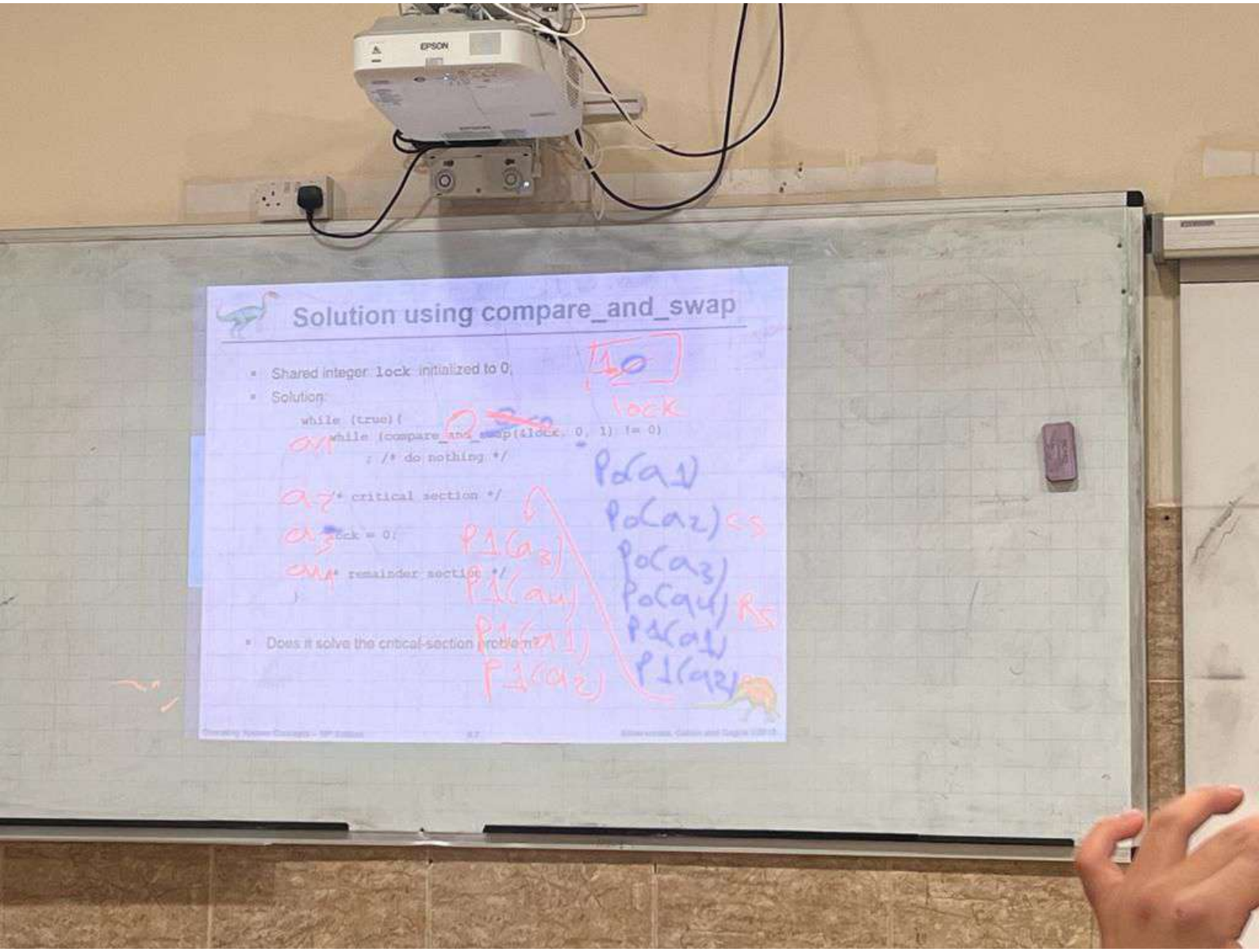


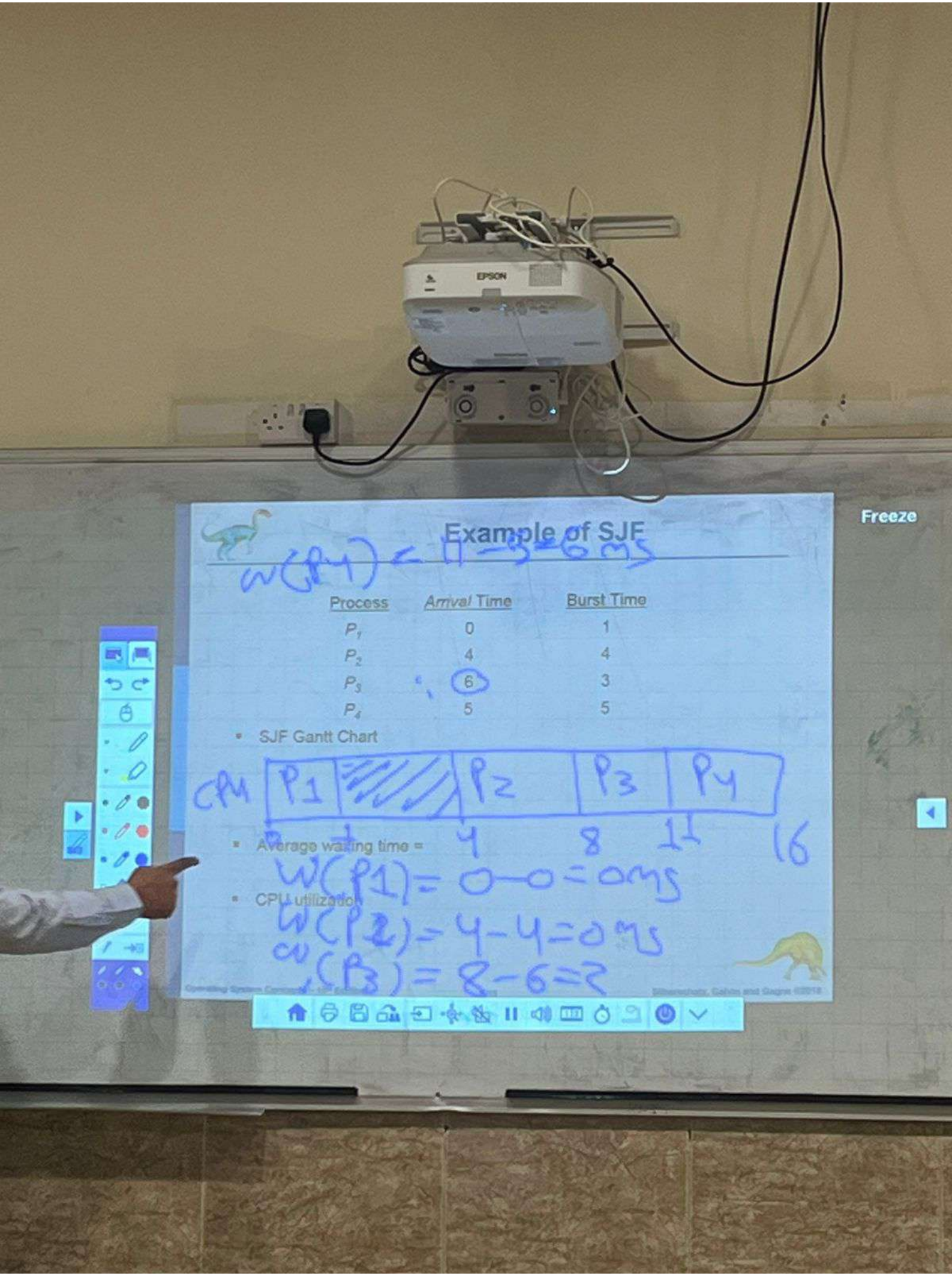
Solution using compare_and_swap

- Shared integer `lock` initialized to 0;
- Solution:

```
while (true){
        while (compare_and_swap(&lock, 0, 1) != 0)
                ; /* do nothing */

        /* critical section */

        lock = 0;

        /* remainder section */
}
```

- Does it solve the critical-section problem?


EPSON
Solution using compare_and_swap
Shared integer lock initialized to 0;
Solution:
while (true){
while (compare_and_swap(&lock, 0, 1) != 0)
; /* do nothing */
/* critical section */
lock = 0;
/* remainder section */
}
Does it solve the critical-section problem?

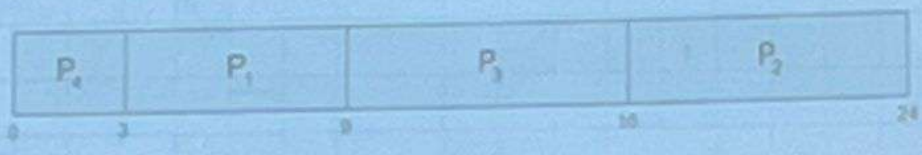

EPSON
Freeze
Example of SJF
W(P4) = 11 - 3 = 6 ms
Process
Arrival Time
Burst Time
P1 0 1
P2 4 4
P3 6 3
P4 5 5
SJF Gantt Chart
CPU
P1
P2
P3
P4
0
1
4
8
11
16
Average waiting time =
W(P1) = 0 - 0 = 0 ms
W(P2) = 4 - 4 = 0 ms
W(P3) = 8 - 6 = 2
CPU utilization

Example of SJF

Process	Burst Time
P_1	6
P_2	8

Turn Around Time: Time taken to complete after arrival. In simple words, it is the difference between the Completion time and the Arrival time.

- SJF scheduling chart

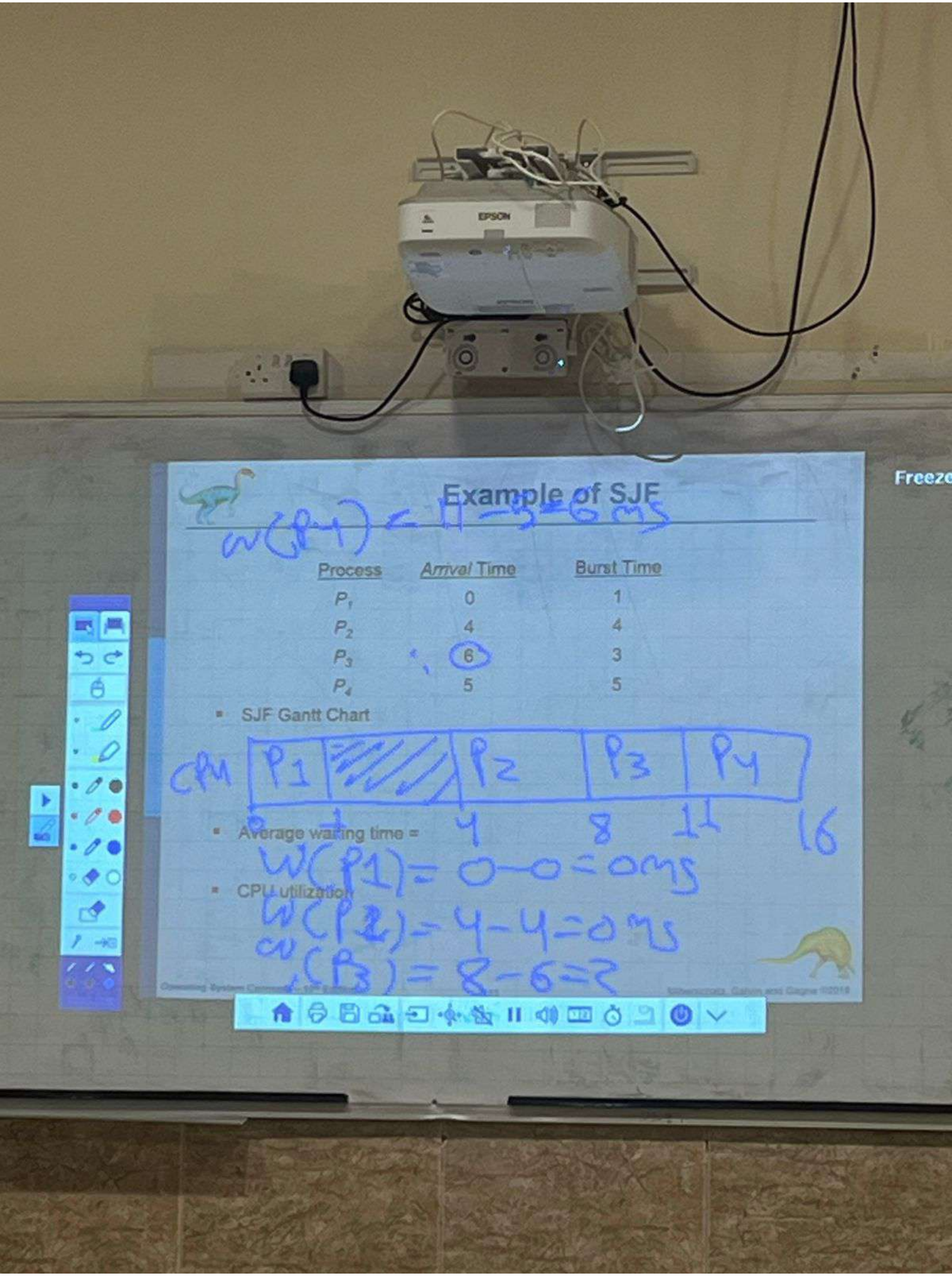


- Average waiting time = (3 + 16 + 9 + 0) / 4 = 7

TAT(

Turn Around Time:

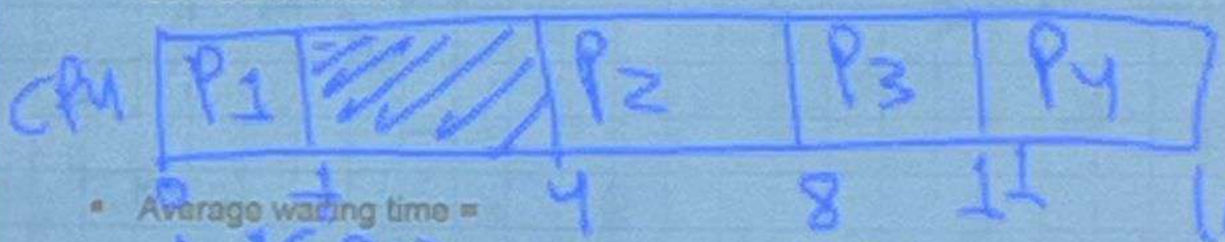
TAT(P1) = 9 - 0 = 9ms


EPSON
Example of SJF
W(P4) = 11 - 5 = 6 ms
Process
Arrival Time
Burst Time
P1 0 1
P2 4 4
P3 6 3
P4 5 5
SJF Gantt Chart
CPU
P1
P2
P3
P4
0
1
4
8
11
16
Average waiting time =
W(P1) = 0 - 0 = 0 ms
W(P2) = 4 - 4 = 0 ms
W(P3) = 8 - 6 = 2
CPU utilization
Freeze

Example of SJF

Process	Arrival Time	Burst Time
P_1	0	7
P_2	4	4
P_3	6	3
P_4	5	5

- SJF Gantt Chart

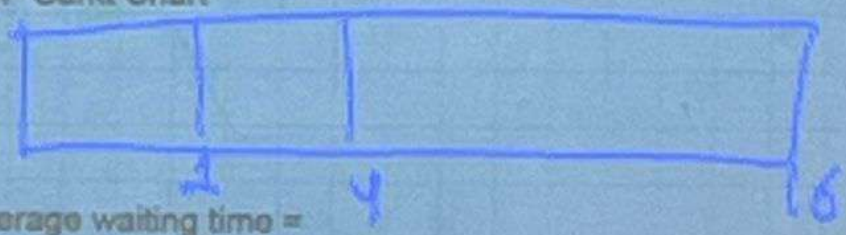


- Average waiting time =
- CPU utilization

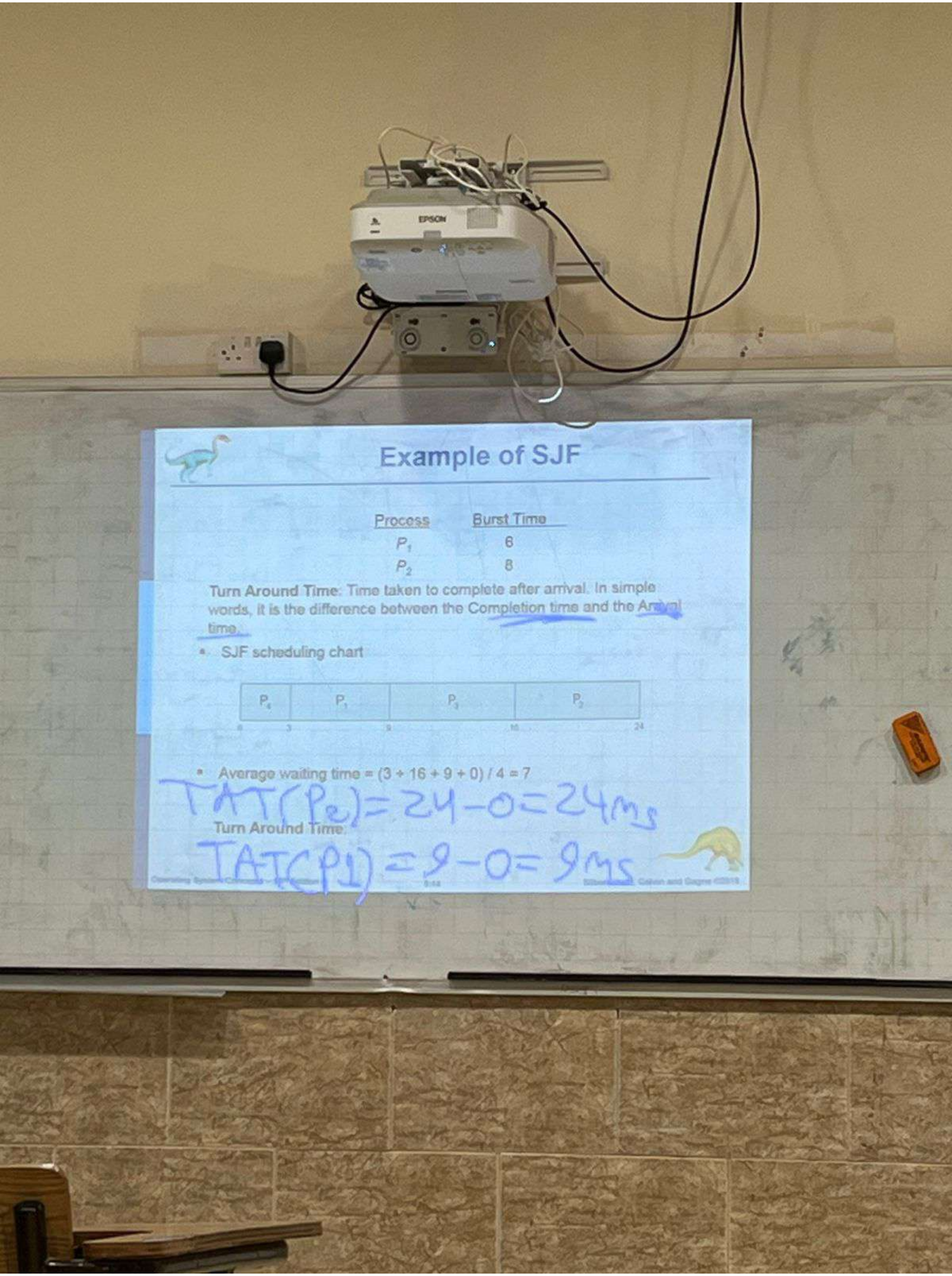
Example of SJF

Process	Arrival Time	Burst Time
P_1	0	1
P_2	4	4
P_3	6	3
P_4	5	5

- SJF Gantt Chart



- Average waiting time =
- CPU utilization $\frac{16-3}{16} * 100\% = 81.8\%$


EPSON
Example of SJF
Process
Burst Time
P_1
6
P_2
8
Turn Around Time: Time taken to complete after arrival. In simple words, it is the difference between the Completion time and the Arrival time.
SJF scheduling chart
P_4
P_1
P_3
P_2
0
3
9
16
24
Average waiting time = (3 + 16 + 9 + 0) / 4 = 7
TAT(P2) = 24 - 0 = 24ms
Turn Around Time:
TAT(P1) = 9 - 0 = 9ms

Example of SJF

Process	Burst Time
P_1	6
P_2	8

Turn Around Time: Time taken to complete after arrival. In simple words, it is the difference between the Completion time and the Arrival time

- SJF scheduling chart

- Average waiting time = (3 + 16 + 9 + 0) / 4 = 7

Turn Around Time:

TAT(P4) = 3 − 0 = 3 ms

TAT(P3) = 16 − 0 = 16 ms

TAT(P2) = 24 − 0 = 24 ms

TAT(P1) = 9 − 0 = 9 ms

Example of SJF

Process	Arrival Time	Burst Time
P_1	0	1
P_2	4	4
P_3	6	3
P_4	5	5

- SJF Gantt Chart
- Average waiting time =
- CPU utilization

TAT(P_1) = 1 - 0 = 1 ms

TAT(P_3) = 11 - 6 = 5 ms

finish

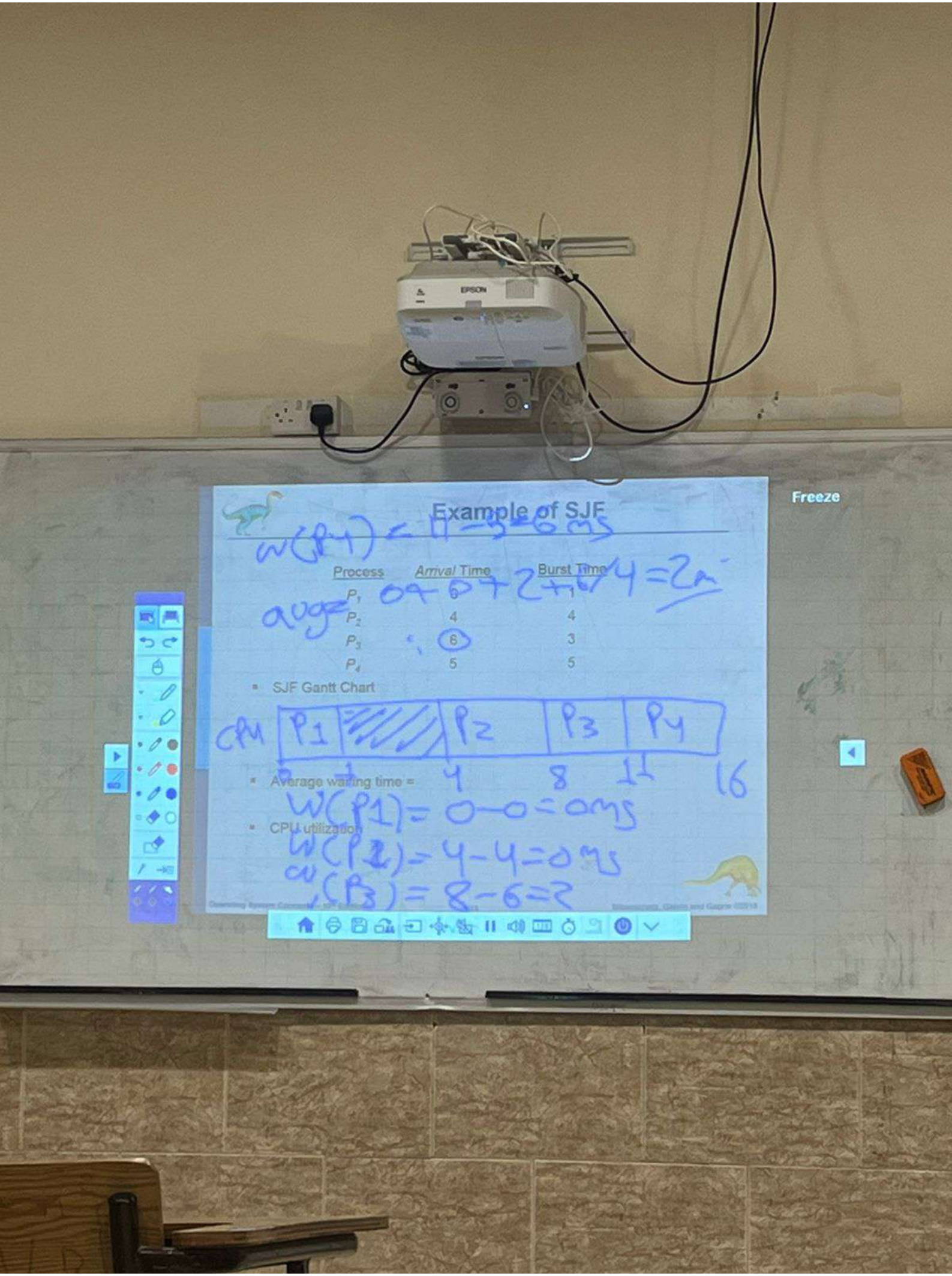
Example of SJF

Process	*Arrival* Time	Burst Time
P_1	0	1
P_2	4	4
P_3	6	3
P_4	5	5

- SJF Gantt Chart

- Average waiting time =

- CPU utilization


EPSON
Freeze
Example of SJF
W(P4) = 11 - 5 = 6 ms
Process
Arrival Time
Burst Time
P1
0
7
P2
4
4
P3
6
3
P4
5
5
avg = 0+7+2+6/4 = 25/4
SJF Gantt Chart
CPU
P1
P2
P3
P4
0
1
4
8
11
16
Average waiting time =
W(P1) = 0 - 0 = 0 ms
W(P2) = 4 - 4 = 0 ms
W(P3) = 8 - 6 = 2
CPU utilization

Bounded-waiting with compare-and-swap

```
waiting[3]=(false)//
lock=0
while (true) {
a1   waiting[i] = true;
a2   key = 1;
a3   while (waiting[i] && key == 1)
a4     key = compare_and_swap(&lock,0,1);
a5   waiting[i] = false;
a6   /* critical section */
a7   j = (i + 1) % n;
a8   while ((j != i) && !waiting[j])
a9     j = (j + 1) % n;
a10  if (j == i)
a11    lock = 0;
     else
a12    waiting[j] = false;
a13  /* remainder section */
```

false	false	false
0	1	2

key