

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA KHOA HỌC - KỸ THUẬT MÁY TÍNH



Report: Assignment 2 - Operating Systems

Topic

Simple Operating System

Lecturer: Nguyễn Hoài Nam
Group 3 Students:
Nguyễn Long Kim - 1812742
Nguyễn Duy Kiên - 1812704
Nguyễn Văn Khang - 1812554
Nguyễn Anh Khoa - 1812649

TP. HỒ CHÍ MINH, THÁNG 6/2020

Task assingment

Members:	Task :	Due date:	Accomplished date:
Nguyễn Duy Kiên	<ul style="list-style-type: none">• Implement functions: enqueue() and dequeue() in queue.c• Draw Gantt chart for sched_0• Answer the question in section Scheduler	24/6/20220	23/6/2020
Nguyễn Văn Khang	<ul style="list-style-type: none">• Implement functions: get_proc() in sched.c• Draw Gantt chart for sched_1• Answer the question in section Scheduler	24/6/20220	23/6/2020
Nguyễn Anh Khoa	<ul style="list-style-type: none">• Implement functions: translate() and get_page_table() in mem.c• Show status of RAM after each allocation and deallocation function call in m0• Answer the question in section Memory Management	24/6/20220	23/6/2020
Nguyễn Long Kim	<ul style="list-style-type: none">• Implement functions: alloc_mem() and free_mem() in mem.c• Show status of RAM after each allocation and deallocation function call in m1• Synchronization for [_ram] access in mem.c• Write report	24/6/20220	23/6/2020

Contents

1	Scheduler	3
1.1	Question & Answer	3
1.2	Gantt chart	3
2	Memory management	7
2.1	Question & Answer	7
2.2	Show status of RAM	7
3	Put it all together	9
3.1	Synchronization	9
3.2	Result of simulation	10

1 Scheduler

1.1 Question & Answer

Question: What is the advantage of using priority feedback queue in comparison with other scheduling algorithms you have learned?

Answer:

- Thời gian đợi trung bình giảm so với giải thuật First Come First Served
- CPU chạy các process theo Round-Robin "style" nên đảm bảo tính công về thời gian thực thi của các process. Không có process nào được chạy quá thời gian mà OS cho phép, từ đó dẫn đến tránh được tình trạng trì hoãn vô hạn định xảy ra ở một số giải thuật định thời khác như: Priority scheduling hay Multilevel Queue Scheduling
- Sử dụng hai hàng đợi *ready_queue* và *run_queue*, các process được chuyển qua lại giữa hai hàng đợi này đến khi các process được hoàn tất, tăng thời gian đáp ứng cho các process do các process có độ ưu tiên thấp đến sau vẫn có thể được thực thi trước các process có độ ưu tiên cao hơn (vì có thể lúc này process có độ ưu tiên cao hơn đang ở *run_queue*).

1.2 Gantt chart

Requirement: Draw Gantt diagram describing how processes are executed by the CPU

1.2.1 Test sched_0

- sched_0 log

```
1 Time slot 0
2   Loaded a process at input/proc/s0, PID: 1
3 Time slot 1
4   CPU 0: Dispatched process 1
5 Time slot 2
6 Time slot 3
7   CPU 0: Put process 1 to run queue
8   CPU 0: Dispatched process 1
9 Time slot 4
10  Loaded a process at input/proc/s1, PID: 2
11 Time slot 5
12  CPU 0: Put process 1 to run queue
13  CPU 0: Dispatched process 2
14 Time slot 6
15 Time slot 7
16  CPU 0: Put process 2 to run queue
17  CPU 0: Dispatched process 2
18 Time slot 8
19 Time slot 9
20  CPU 0: Put process 2 to run queue
21  CPU 0: Dispatched process 1
22 Time slot 10
23 Time slot 11
24  CPU 0: Put process 1 to run queue
25  CPU 0: Dispatched process 2
26 Time slot 12
27 Time slot 13
28  CPU 0: Put process 2 to run queue
29  CPU 0: Dispatched process 1
30 Time slot 14
31 Time slot 15
32  CPU 0: Put process 1 to run queue
33  CPU 0: Dispatched process 2
34 Time slot 16
35  CPU 0: Processed 2 has finished
36  CPU 0: Dispatched process 1
37 Time slot 17
38 Time slot 18
39  CPU 0: Put process 1 to run queue
40  CPU 0: Dispatched process 1
```

```

41 Time slot 19
42 Time slot 20
43     CPU 0: Put process 1 to run queue
44     CPU 0: Dispatched process 1
45 Time slot 21
46 Time slot 22
47     CPU 0: Put process 1 to run queue
48     CPU 0: Dispatched process 1
49 Time slot 23
50     CPU 0: Processed 1 has finished
51     CPU 0 stopped
52
53 MEMORY CONTENT:

```

- Gantt chart

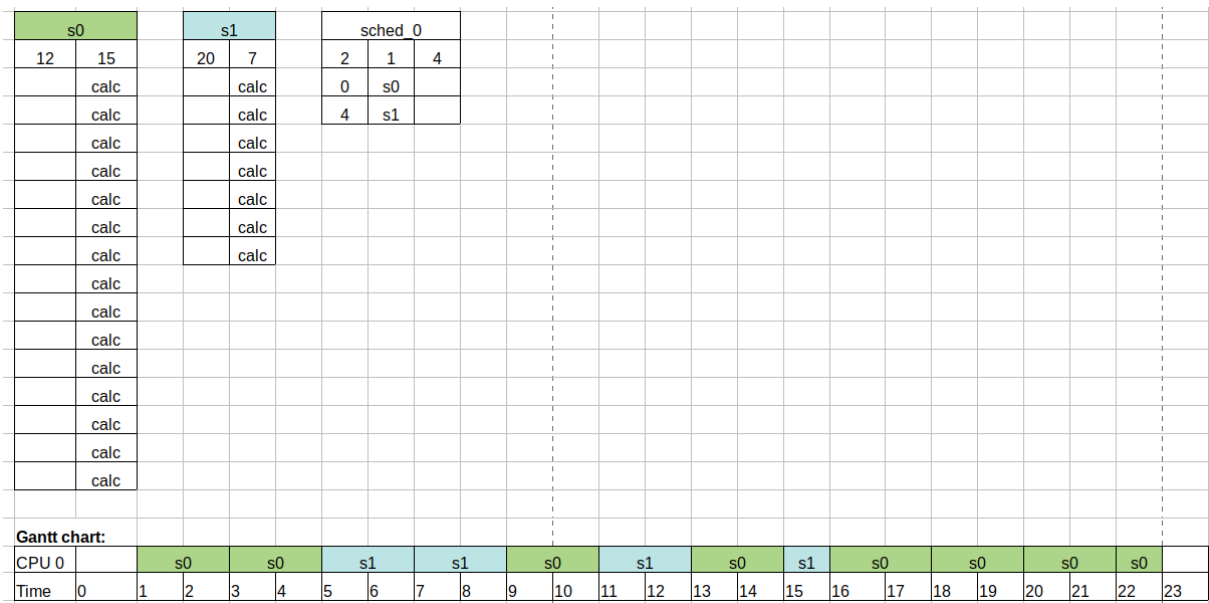


Figure 1: Gantt chart CPU thực thi các process trong sched 0

1.2.2 Test sched 1

- sched 1 log

```

1 Time slot      0
2           Loaded a process at input/proc/s0, PID: 1
3 Time slot      1
4           CPU 0: Dispatched process 1
5 Time slot      2
6 Time slot      3
7           CPU 0: Put process 1 to run queue
8           CPU 0: Dispatched process 1
9 Time slot      4
10          Loaded a process at input/proc/s1, PID: 2
11 Time slot      5
12          CPU 0: Put process 1 to run queue
13          CPU 0: Dispatched process 2
14 Time slot      6
15          Loaded a process at input/proc/s2, PID: 3
16 Time slot      7
17          CPU 0: Put process 2 to run queue
18          CPU 0: Dispatched process 3
19          Loaded a process at input/proc/s3, PID: 4
20 Time slot      8
21 Time slot      9
22          CPU 0: Put process 3 to run queue

```

```
23         CPU 0: Dispatched process 4
24 Time slot 10
25 Time slot 11
26         CPU 0: Put process 4 to run queue
27         CPU 0: Dispatched process 2
28 Time slot 12
29 Time slot 13
30         CPU 0: Put process 2 to run queue
31         CPU 0: Dispatched process 3
32 Time slot 14
33 Time slot 15
34         CPU 0: Put process 3 to run queue
35         CPU 0: Dispatched process 1
36 Time slot 16
37 Time slot 17
38         CPU 0: Put process 1 to run queue
39         CPU 0: Dispatched process 4
40 Time slot 18
41 Time slot 19
42         CPU 0: Put process 4 to run queue
43         CPU 0: Dispatched process 2
44 Time slot 20
45 Time slot 21
46         CPU 0: Put process 2 to run queue
47         CPU 0: Dispatched process 3
48 Time slot 22
49 Time slot 23
50         CPU 0: Put process 3 to run queue
51         CPU 0: Dispatched process 1
52 Time slot 24
53 Time slot 25
54         CPU 0: Put process 1 to run queue
55         CPU 0: Dispatched process 4
56 Time slot 26
57 Time slot 27
58         CPU 0: Put process 4 to run queue
59         CPU 0: Dispatched process 2
60 Time slot 28
61         CPU 0: Processed 2 has finished
62         CPU 0: Dispatched process 3
63 Time slot 29
64 Time slot 30
65         CPU 0: Put process 3 to run queue
66         CPU 0: Dispatched process 1
67 Time slot 31
68 Time slot 32
69         CPU 0: Put process 1 to run queue
70         CPU 0: Dispatched process 4
71 Time slot 33
72 Time slot 34
73         CPU 0: Put process 4 to run queue
74         CPU 0: Dispatched process 3
75 Time slot 35
76 Time slot 36
77         CPU 0: Put process 3 to run queue
78         CPU 0: Dispatched process 1
79 Time slot 37
80 Time slot 38
81         CPU 0: Put process 1 to run queue
82         CPU 0: Dispatched process 4
83 Time slot 39
84 Time slot 40
85         CPU 0: Put process 4 to run queue
86         CPU 0: Dispatched process 3
87 Time slot 41
88 Time slot 42
89         CPU 0: Processed 3 has finished
90         CPU 0: Dispatched process 1
91 Time slot 43
92 Time slot 44
93         CPU 0: Put process 1 to run queue
94         CPU 0: Dispatched process 4
95 Time slot 45
```

```

96          CPU 0: Processed 4 has finished
97          CPU 0: Dispatched process 1
98 Time slot 46
99          CPU 0: Processed 1 has finished
100         CPU 0 stopped
101
102 MEMORY CONTENT:

```

- Gantt chart

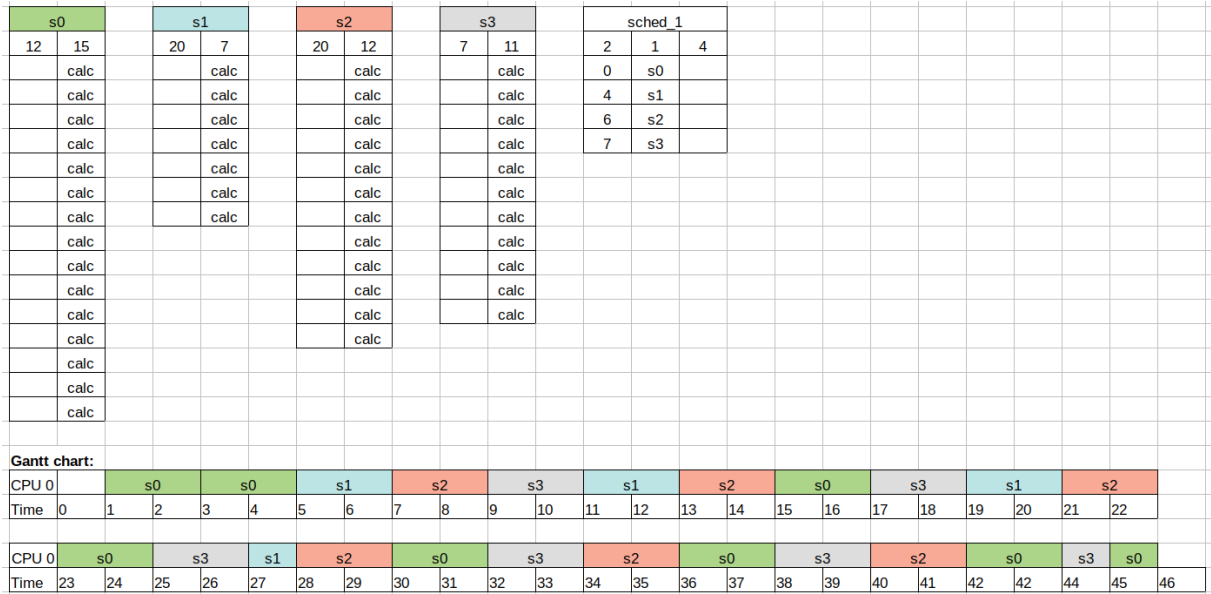


Figure 2: Gantt chart CPU thực thi các process trong sched_1

2 Memory management

2.1 Question & Answer

Question: What is the advantage and disadvantage of segmentation with paging?

Answer:

- **Advantage:**

- Tiết kiệm bộ nhớ, sử dụng bộ nhớ hiệu quả, khắc phục việc kích thước bảng phân trang quá lớn
- Khắc phục được phân mảnh ngoại

- **Disadvantage:**

- Chưa khắc phục được hiện tượng phân mảnh nội

2.2 Show status of RAM

Requirement: Show the status of RAM after each memory allocation and deallocation function call

2.2.1 Test *m0*

- Below is content of file *m0*

```
1 1 7
2 alloc 13535 0
3 alloc 1568 1
4 free 0
5 alloc 1386 2
6 alloc 4564 4
7 write 102 1 20
8 write 21 2 1000
```

- Status of RAM after each allocation and deallocation function call in *m0*

```
1 alloc 13535 0:
2 000: 00000-003ff - PID: 01 (idx 000, nxt: 001)
3 001: 00400-007ff - PID: 01 (idx 001, nxt: 002)
4 002: 00800-00bff - PID: 01 (idx 002, nxt: 003)
5 003: 00c00-00fff - PID: 01 (idx 003, nxt: 004)
6 004: 01000-013ff - PID: 01 (idx 004, nxt: 005)
7 005: 01400-017ff - PID: 01 (idx 005, nxt: 006)
8 006: 01800-01bff - PID: 01 (idx 006, nxt: 007)
9 007: 01c00-01fff - PID: 01 (idx 007, nxt: 008)
10 008: 02000-023ff - PID: 01 (idx 008, nxt: 009)
11 009: 02400-027ff - PID: 01 (idx 009, nxt: 010)
12 010: 02800-02bff - PID: 01 (idx 010, nxt: 011)
13 011: 02c00-02fff - PID: 01 (idx 011, nxt: 012)
14 012: 03000-033ff - PID: 01 (idx 012, nxt: 013)
15 013: 03400-037ff - PID: 01 (idx 013, nxt: -01)
16
17 alloc 1568 1:
18 000: 00000-003ff - PID: 01 (idx 000, nxt: 001)
19 001: 00400-007ff - PID: 01 (idx 001, nxt: 002)
20 002: 00800-00bff - PID: 01 (idx 002, nxt: 003)
21 003: 00c00-00fff - PID: 01 (idx 003, nxt: 004)
22 004: 01000-013ff - PID: 01 (idx 004, nxt: 005)
23 005: 01400-017ff - PID: 01 (idx 005, nxt: 006)
24 006: 01800-01bff - PID: 01 (idx 006, nxt: 007)
25 007: 01c00-01fff - PID: 01 (idx 007, nxt: 008)
26 008: 02000-023ff - PID: 01 (idx 008, nxt: 009)
27 009: 02400-027ff - PID: 01 (idx 009, nxt: 010)
28 010: 02800-02bff - PID: 01 (idx 010, nxt: 011)
29 011: 02c00-02fff - PID: 01 (idx 011, nxt: 012)
30 012: 03000-033ff - PID: 01 (idx 012, nxt: 013)
31 013: 03400-037ff - PID: 01 (idx 013, nxt: -01)
```



```
32 014: 03800-03bff - PID: 01 (idx 000, nxt: 015)
33 015: 03c00-03fff - PID: 01 (idx 001, nxt: -01)
34
35 free 0:
36 014: 03800-03bff - PID: 01 (idx 000, nxt: 015)
37 015: 03c00-03fff - PID: 01 (idx 001, nxt: -01)
38
39 alloc 1386 2:
40 000: 00000-003ff - PID: 01 (idx 000, nxt: 001)
41 001: 00400-007ff - PID: 01 (idx 001, nxt: -01)
42 014: 03800-03bff - PID: 01 (idx 000, nxt: 015)
43 015: 03c00-03fff - PID: 01 (idx 001, nxt: -01)
44
45 alloc 4564 4:
46 000: 00000-003ff - PID: 01 (idx 000, nxt: 001)
47 001: 00400-007ff - PID: 01 (idx 001, nxt: -01)
48 002: 00800-00bff - PID: 01 (idx 000, nxt: 003)
49 003: 00c00-00fff - PID: 01 (idx 001, nxt: 004)
50 004: 01000-013ff - PID: 01 (idx 002, nxt: 005)
51 005: 01400-017ff - PID: 01 (idx 003, nxt: 006)
52 006: 01800-01bff - PID: 01 (idx 004, nxt: -01)
53 014: 03800-03bff - PID: 01 (idx 000, nxt: 015)
54 015: 03c00-03fff - PID: 01 (idx 001, nxt: -01)
55
```

2.2.2 Test *m1*

- Below is content of file *m1*

```
1 1 8
2 alloc 13535 0
3 alloc 1568 1
4 free 0
5 alloc 1386 2
6 alloc 4564 4
7 free 2
8 free 4
9 free 1
```

- Status of RAM after each allocation and deallocation function call in *m1*

```
1 alloc 13535 0:
2 000: 00000-003ff - PID: 01 (idx 000, nxt: 001)
3 001: 00400-007ff - PID: 01 (idx 001, nxt: 002)
4 002: 00800-00bff - PID: 01 (idx 002, nxt: 003)
5 003: 00c00-00fff - PID: 01 (idx 003, nxt: 004)
6 004: 01000-013ff - PID: 01 (idx 004, nxt: 005)
7 005: 01400-017ff - PID: 01 (idx 005, nxt: 006)
8 006: 01800-01bff - PID: 01 (idx 006, nxt: 007)
9 007: 01c00-01fff - PID: 01 (idx 007, nxt: 008)
10 008: 02000-023ff - PID: 01 (idx 008, nxt: 009)
11 009: 02400-027ff - PID: 01 (idx 009, nxt: 010)
12 010: 02800-02bff - PID: 01 (idx 010, nxt: 011)
13 011: 02c00-02fff - PID: 01 (idx 011, nxt: 012)
14 012: 03000-033ff - PID: 01 (idx 012, nxt: 013)
15 013: 03400-037ff - PID: 01 (idx 013, nxt: -01)
16
17 alloc 1568 1:
18 000: 00000-003ff - PID: 01 (idx 000, nxt: 001)
19 001: 00400-007ff - PID: 01 (idx 001, nxt: 002)
20 002: 00800-00bff - PID: 01 (idx 002, nxt: 003)
21 003: 00c00-00fff - PID: 01 (idx 003, nxt: 004)
22 004: 01000-013ff - PID: 01 (idx 004, nxt: 005)
23 005: 01400-017ff - PID: 01 (idx 005, nxt: 006)
24 006: 01800-01bff - PID: 01 (idx 006, nxt: 007)
25 007: 01c00-01fff - PID: 01 (idx 007, nxt: 008)
26 008: 02000-023ff - PID: 01 (idx 008, nxt: 009)
27 009: 02400-027ff - PID: 01 (idx 009, nxt: 010)
28 010: 02800-02bff - PID: 01 (idx 010, nxt: 011)
```

```
29 011: 02c00-02fff - PID: 01 (idx 011, nxt: 012)
30 012: 03000-033ff - PID: 01 (idx 012, nxt: 013)
31 013: 03400-037ff - PID: 01 (idx 013, nxt: -01)
32 014: 03800-03bff - PID: 01 (idx 000, nxt: 015)
33 015: 03c00-03fff - PID: 01 (idx 001, nxt: -01)
34
35 free 0:
36 014: 03800-03bff - PID: 01 (idx 000, nxt: 015)
37 015: 03c00-03fff - PID: 01 (idx 001, nxt: -01)
38
39 alloc 1386 2:
40 000: 00000-003ff - PID: 01 (idx 000, nxt: 001)
41 001: 00400-007ff - PID: 01 (idx 001, nxt: -01)
42 014: 03800-03bff - PID: 01 (idx 000, nxt: 015)
43 015: 03c00-03fff - PID: 01 (idx 001, nxt: -01)
44
45 alloc 4564 4
46 000: 00000-003ff - PID: 01 (idx 000, nxt: 001)
47 001: 00400-007ff - PID: 01 (idx 001, nxt: -01)
48 002: 00800-00bff - PID: 01 (idx 000, nxt: 003)
49 003: 00c00-00fff - PID: 01 (idx 001, nxt: 004)
50 004: 01000-013ff - PID: 01 (idx 002, nxt: 005)
51 005: 01400-017ff - PID: 01 (idx 003, nxt: 006)
52 006: 01800-01bff - PID: 01 (idx 004, nxt: -01)
53 014: 03800-03bff - PID: 01 (idx 000, nxt: 015)
54 015: 03c00-03fff - PID: 01 (idx 001, nxt: -01)
55
56 free 2:
57 002: 00800-00bff - PID: 01 (idx 000, nxt: 003)
58 003: 00c00-00fff - PID: 01 (idx 001, nxt: 004)
59 004: 01000-013ff - PID: 01 (idx 002, nxt: 005)
60 005: 01400-017ff - PID: 01 (idx 003, nxt: 006)
61 006: 01800-01bff - PID: 01 (idx 004, nxt: -01)
62 014: 03800-03bff - PID: 01 (idx 000, nxt: 015)
63 015: 03c00-03fff - PID: 01 (idx 001, nxt: -01)
64
65 free 4:
66 014: 03800-03bff - PID: 01 (idx 000, nxt: 015)
67 015: 03c00-03fff - PID: 01 (idx 001, nxt: -01)
68
69 free 1:
```

3 Put it all together

3.1 Synchronization

Trong file *mem.c*, hai hàm *read_mem()* và *write_mem()* truy cập vào *[_ram]* có thể xảy ra bất đồng bộ nên ta cần thêm một *mutex_lock: ram_lock* để giải quyết, khi đó hai hàm trên được chỉnh lại như sau:

- Hàm *read_mem()*

```
1 int read_mem(addr_t address, struct pcb_t * proc, BYTE * data) {
2     addr_t physical_addr;
3     if (translate(address, &physical_addr, proc)) {
4         pthread_mutex_lock(&ram_lock); // add
5         *data = _ram[physical_addr];
6         pthread_mutex_unlock(&ram_lock); // add
7         return 0;
8     } else {
9         return 1;
10    }
11 }
```

- Hàm *write_mem()*

```
1 int write_mem(addr_t address, struct pcb_t * proc, BYTE data) {
```

```
2  addr_t physical_addr;  
3  if (translate(address, &physical_addr, proc)) {  
4      pthread_mutex_lock(&ram_lock); // add  
5      _ram[physical_addr] = data;  
6      pthread_mutex_unlock(&ram_lock); // add  
7      return 0;  
8  } else {  
9      return 1;  
10 }  
11 }
```

3.2 Result of simulation

• os_0 log

```
1 Time slot 0  
2   Loaded a process at input/proc/p0, PID: 1  
3 Time slot 1  
4   CPU 0: Dispatched process 1  
5 Time slot 2  
6   Loaded a process at input/proc/p1, PID: 2  
7 Time slot 3  
8   CPU 1: Dispatched process 2  
9 Time slot 4  
10 Time slot 5  
11 Time slot 6  
12 Time slot 7  
13   CPU 0: Put process 1 to run queue  
14   CPU 0: Dispatched process 1  
15 Time slot 8  
16 Time slot 9  
17   CPU 1: Put process 2 to run queue  
18   CPU 1: Dispatched process 2  
19 Time slot 10  
20 Time slot 11  
21 Time slot 12  
22   CPU 0: Processed 1 has finished  
23   CPU 0 stopped  
24 Time slot 13  
25   CPU 1: Processed 2 has finished  
26   CPU 1 stopped  
27  
28 MEMORY CONTENT:  
29 000: 00000-003ff - PID: 02 (idx 000, nxt: 001)  
30 001: 00400-007ff - PID: 02 (idx 001, nxt: 007)  
31 007: 01c00-01fff - PID: 02 (idx 002, nxt: 008)  
32   01de7: 0a  
33 008: 02000-023ff - PID: 02 (idx 003, nxt: 009)  
34 009: 02400-027ff - PID: 02 (idx 004, nxt: -01)  
35 010: 02800-02bff - PID: 01 (idx 000, nxt: -01)  
36   02814: 64  
37 015: 03c00-03fff - PID: 02 (idx 000, nxt: 016)  
38 016: 04000-043ff - PID: 02 (idx 001, nxt: 017)  
39 017: 04400-047ff - PID: 02 (idx 002, nxt: 018)  
40 018: 04800-04bff - PID: 02 (idx 003, nxt: -01)
```

• os_1 log

```
1 Time slot 0  
2   Loaded a process at input/proc/p0, PID: 1  
3 Time slot 1  
4   CPU 2: Dispatched process 1  
5 Time slot 2  
6   Loaded a process at input/proc/s3, PID: 2  
7 Time slot 3  
8   CPU 0: Dispatched process 2  
9   CPU 2: Put process 1 to run queue  
10   CPU 2: Dispatched process 1  
11   Loaded a process at input/proc/m1, PID: 3
```

```
12 Time slot 4
13     CPU 1: Dispatched process 3
14     CPU 2: Put process 1 to run queue
15     CPU 2: Dispatched process 1
16     CPU 0: Put process 2 to run queue
17     CPU 0: Dispatched process 2
18 Time slot 5
19     Loaded a process at input/proc/s2, PID: 4
20 Time slot 6
21     CPU 1: Put process 3 to run queue
22     CPU 1: Dispatched process 4
23     CPU 3: Dispatched process 3
24     CPU 0: Put process 2 to run queue
25     Loaded a process at input/proc/m0, PID: 5
26     CPU 2: Put process 1 to run queue
27     CPU 2: Dispatched process 5
28 Time slot 7
29     CPU 0: Dispatched process 2
30 Time slot 8
31     CPU 1: Put process 4 to run queue
32     CPU 1: Dispatched process 4
33     Loaded a process at input/proc/p1, PID: 6
34     CPU 2: Put process 5 to run queue
35     CPU 0: Put process 2 to run queue
36     CPU 0: Dispatched process 6
37     CPU 2: Dispatched process 1
38 Time slot 9
39     CPU 3: Put process 3 to run queue
40     CPU 3: Dispatched process 2
41 Time slot 10
42     CPU 1: Put process 4 to run queue
43     CPU 1: Dispatched process 3
44     CPU 3: Put process 2 to run queue
45     CPU 2: Put process 1 to run queue
46     CPU 2: Dispatched process 4
47     CPU 3: Dispatched process 5
48     Loaded a process at input/proc/s0, PID: 7
49     CPU 0: Put process 6 to run queue
50     CPU 0: Dispatched process 7
51 Time slot 11
52 Time slot 12
53     CPU 1: Put process 3 to run queue
54     CPU 1: Dispatched process 2
55     CPU 3: Put process 5 to run queue
56 Time slot 13
57     CPU 0: Put process 7 to run queue
58     CPU 0: Dispatched process 7
59     CPU 2: Put process 4 to run queue
60     CPU 2: Dispatched process 3
61     CPU 3: Dispatched process 1
62 Time slot 14
63     CPU 1: Put process 2 to run queue
64     CPU 1: Dispatched process 5
65     CPU 3: Put process 1 to run queue
66 Time slot 15
67     CPU 3: Dispatched process 6
68     CPU 0: Put process 7 to run queue
69     CPU 0: Dispatched process 4
70     CPU 2: Processed 3 has finished
71     CPU 2: Dispatched process 7
72     Loaded a process at input/proc/s1, PID: 8
73 Time slot 16
74     CPU 1: Put process 5 to run queue
75     CPU 1: Dispatched process 8
76     CPU 3: Put process 6 to run queue
77 Time slot 17
78     CPU 0: Put process 4 to run queue
79     CPU 0: Dispatched process 1
80     CPU 3: Dispatched process 2
81     CPU 2: Put process 7 to run queue
82     CPU 2: Dispatched process 4
83     CPU 3: Processed 2 has finished
84     CPU 1: Put process 8 to run queue
```

```
85         CPU 1: Dispatched process 6
86         CPU 3: Dispatched process 7
87         CPU 0: Processed 1 has finished
88         CPU 0: Dispatched process 5
89 Time slot 18
90         CPU 2: Put process 4 to run queue
91         CPU 0: Processed 5 has finished
92         CPU 0: Dispatched process 4
93 Time slot 19
94         CPU 2: Dispatched process 8
95         CPU 3: Put process 7 to run queue
96         CPU 3: Dispatched process 7
97 Time slot 20
98         CPU 1: Put process 6 to run queue
99         CPU 1: Dispatched process 6
100        CPU 2: Put process 8 to run queue
101        CPU 0: Processed 4 has finished
102        CPU 0 stopped
103 Time slot 21
104        CPU 2: Dispatched process 8
105        CPU 3: Put process 7 to run queue
106        CPU 3: Dispatched process 7
107        CPU 1: Put process 6 to run queue
108        CPU 1: Dispatched process 6
109 Time slot 22
110 Time slot 23
111        CPU 2: Put process 8 to run queue
112        CPU 2: Dispatched process 8
113 Time slot 24
114        CPU 3: Put process 7 to run queue
115        CPU 3: Dispatched process 7
116        CPU 1: Processed 6 has finished
117        CPU 1 stopped
118        CPU 2: Processed 8 has finished
119        CPU 2 stopped
120 Time slot 25
121 Time slot 26
122        CPU 3: Put process 7 to run queue
123        CPU 3: Dispatched process 7
124 Time slot 27
125        CPU 3: Processed 7 has finished
126        CPU 3 stopped
127
128 MEMORY CONTENT:
129 000: 00000-003ff - PID: 05 (idx 000, nxt: 001)
130      003e8: 15
131 001: 00400-007ff - PID: 05 (idx 001, nxt: -01)
132 002: 00800-00bff - PID: 05 (idx 000, nxt: 003)
133 003: 00c00-00fff - PID: 05 (idx 001, nxt: 004)
134 004: 01000-013ff - PID: 05 (idx 002, nxt: 005)
135 005: 01400-017ff - PID: 05 (idx 003, nxt: 006)
136 006: 01800-01bff - PID: 05 (idx 004, nxt: -01)
137 011: 02c00-02fff - PID: 06 (idx 000, nxt: 012)
138 012: 03000-033ff - PID: 06 (idx 001, nxt: 013)
139 013: 03400-037ff - PID: 06 (idx 002, nxt: 014)
140 014: 03800-03bff - PID: 06 (idx 003, nxt: -01)
141 019: 04c00-04fff - PID: 01 (idx 000, nxt: -01)
142      04c14: 64
143 024: 06000-063ff - PID: 05 (idx 000, nxt: 025)
144      06014: 66
145 025: 06400-067ff - PID: 05 (idx 001, nxt: -01)
146 026: 06800-06bff - PID: 06 (idx 000, nxt: 027)
147 027: 06c00-06fff - PID: 06 (idx 001, nxt: 028)
148 028: 07000-073ff - PID: 06 (idx 002, nxt: 029)
149      071e7: 0a
150 029: 07400-077ff - PID: 06 (idx 003, nxt: 030)
151 030: 07800-07bff - PID: 06 (idx 004, nxt: -01)
```