Selected solutions for

TDDD07
Real-time Systems

Klervie Toczé
Massimiliano Raciti
Jordi Cucurull
Simin Nadjm-Tehrani

Scheduling and resource sharing

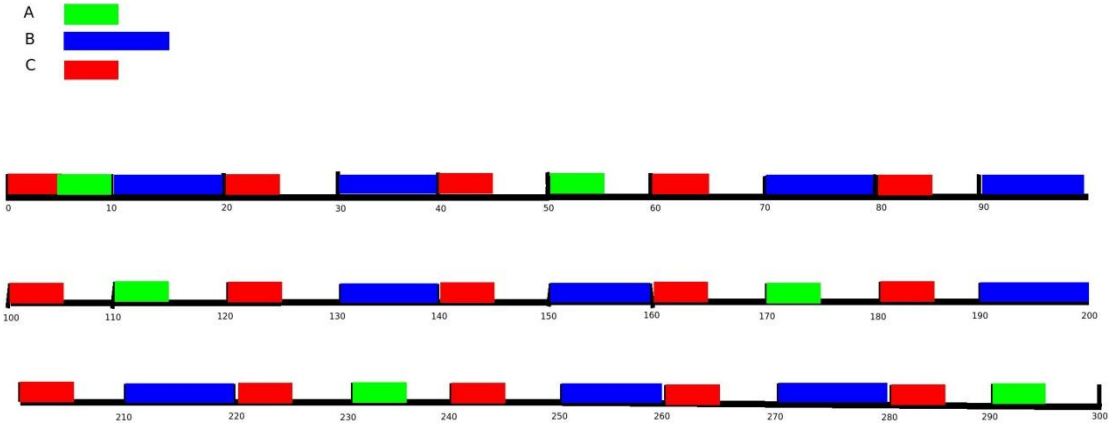
Real-Time Systems Laboratory
Department of Computer and Information Science
Linköping University, Sweden

September 2024

Suggested Solutions

1. Scheduling

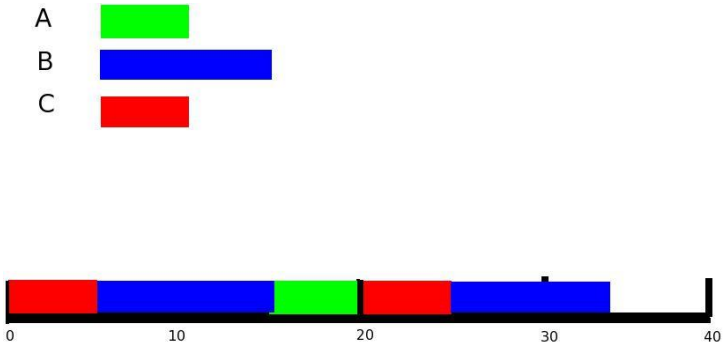
Q1.1:
The task set is schedulable using cyclic scheduling, but the tasks suffer jitter.
Here is an example of scheduling, in which you can see that tasks are affected by jitter
Major cycle: lcm(50,30,20) = 300ms
Minor cycle: gcd(50,30,20)=10ms



You can try to change the order of execution of the tasks, but the jitter still remains. One of the possible ways to schedule this task set without jitter is by reducing the periods, thus executing tasks more frequently than required (see Example (3.2), i.e. Alternative 2 in the slides of Lecture 2).

Task	*New Period (ms)*
A	40
B	20
C	20

Major cycle: lcm(40,20,20) = 40ms
Minor cycle: gcd(40,20,20)=20ms



Q1.2:

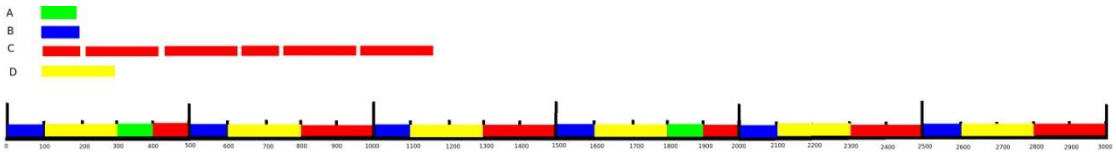
a)

Task	*Period*	*WCET*
A (Trajectory follower)	1500	100
B (Sensor & measurement)	500	100
C (Disk storage)	3000	1000
D (Ground communication)	500	200

Major cycle: lcm(1500,500,3000,500) = 3000ms
Minor cycle: gcd(1500,500,3000,500)=500ms

The task set is not schedulable. This is due to the disk storage task, that cannot be fitted in any place inside the major cycle. The task set becomes schedulable if we assume that the disk storage task can be divided into 6 processes as following:

C1=100ms,C2=200ms,C3=200ms,C4=100ms,C5=200ms, C6=200ms.



b) The new task set is:

Task	*Period*	*WCET*
Trajectory follower	1500	100
Sensor & measurement	1000 (incr. from 500)	100
Disk storage	3000	1000
Ground communication	500	100 (reduc. from 200)

RMS:

$$U = \frac{100}{1500} + \frac{100}{1000} + \frac{1000}{3000} + \frac{100}{500} = \frac{200 + 300 + 1000 + 600}{3000} = \frac{2100}{3000} = 70\%$$

G for 4 tasks is:

$$G = n(2^{\frac{1}{n}} - 1) = 4(2^{\frac{1}{4}} - 1) \approx 75\%$$

The task set is then schedulable using RMS.
However, the jitter requirements are not met. Since the trajectory follower is not the process with higher priority, it can be preempted by higher priority processes, thus causing jitter.

c) This phenomenon is the priority inversion. When the sensor recording task is blocked waiting for the busy resource hold by the disk storage (which has the lowest priority), middle priority processes can preempt this one causing the sensor recording to wait for them as well.

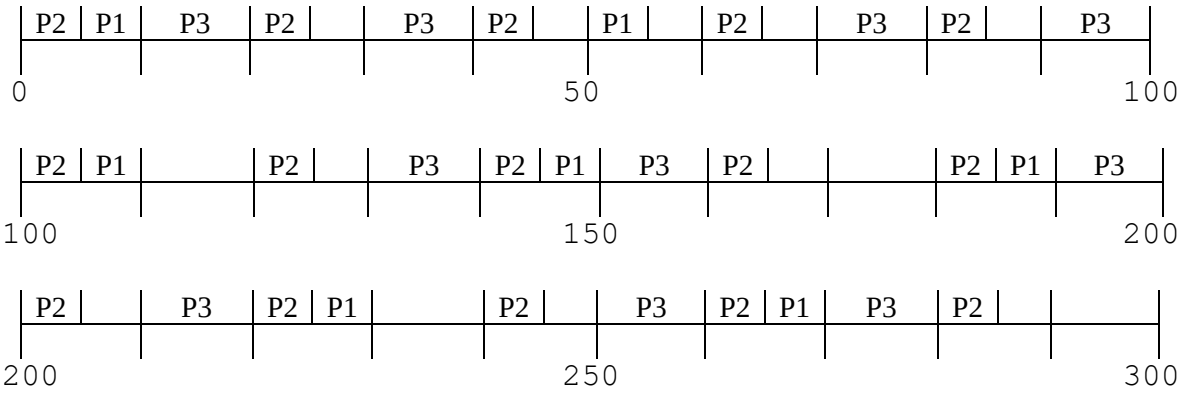
This problem can be solved using the priority inheritance protocol or the priority ceiling protocol, although the second one is preferable since it prevents deadlocks.

Q1.3:

1) According to the description we have the following task set:

Process	Period	WCET
P1 – Stabiliser	50ms	5ms
P2 – Star follower	20ms	5ms
P3 – Energy manager	30ms	10ms

Minor cycle: gcd (50, 20, 30) = 10ms
Major cycle: lcm (50, 20, 30) = 300 ms



Note: To guarantee that P1 provides an output every 50ms maximum, an extra execution of P1 needs to be added.

2)

Process	Period	WCET	Shares	Critical Section Time	Priority (π)
P1 – Stabiliser	50ms	5ms	R2 – Position	1ms	3
P2 – Star follower	20ms	5ms	R1 – Memory	2ms	5 (Highest)
P3 – Energy manager	30ms	10ms	-	-	4
P4 – Picture storage	50ms	-	R1 – Memory	2ms	2
P5 - Communication	100ms	-	R2 – Position	1ms	1(Lowest)

Ceiling (R1) = max (π2, π4) = 5
Ceiling (R2) = max (π1, π5) = 3

The blocking times are calculated. In order to simplify the process it is recommended to go through the lower to higher priority processes. Let $lp\{Z\}$ denote the set of processes with lower priority than Z:

B_{P5}: There is no process with lower priority $lp\{P5\} = \{\}$. No process can block P5 then $B_{P5} = 0ms$.

B_{P4}: In this case $lp\{P4\} = \{P5\}$. We have to check if P4 can be blocked by the lower priority process P5:

(i) Process P5 locks R2 and $Ceiling(R2) = 3 \geq \pi_{P4} = 2$. Process P5 can block P4.

Process P4 can be blocked by process P5. The blocking time is $B_{P4} = max(\, t_{R2,P5}) = 1ms$.

B_{P1}: In this case $lp\{P1\} = \{P4, P5\}$. We have to check if P1 can be blocked by any of these two lower priority processes:

(i) Process P5 locks R2 and $Ceiling(R2) = 3 \geq \pi_{P1} = 3$. Process P5 can block P1.
(ii) Process P4 locks R1 and $Ceiling(R1) = 5 \geq \pi_{P1} = 3$. Process P4 can block P1.

Process P1 can be blocked by processes P5 and P4. Then, the blocking time is $B_{P1} = max(\, t_{R2,P5}, t_{R1,P4}) = 2ms$.

B_{P3}: In this case $lp\{P3\} = \{P5, P4, P1\}$. We have to check if P3 can be blocked by any of these lower priority processes:

(i) Process P5 locks R2 and $Ceiling(R2) = 3 \ngeq \pi_{P3} = 4$. Process P5 cannot block P3.
(ii) Process P4 locks R1 and $Ceiling(R1) = 5 \geq \pi_{P3} = 4$. Process P4 can block P3.
(iii) Process P1 locks R1 and $Ceiling(R2) = 3 \ngeq \pi_{P3} = 4$. Process P1 cannot block P3.

Process P3 can only be blocked by process P4. Then, the blocking time is $B_{P3} = max(\, t_{R1,P4}) = 2ms$.

B_{P2}: In this case $lp\{P2\} = \{P5, P4, P1, P3\}$. We have to check if P2 can be blocked by any of these lower priority processes:

(i) Process P5 locks R2 and $Ceiling(R2) = 3 \ngeq \pi_{P2} = 5$. Process P5 cannot block P2.
(ii) Process P4 locks R1 and $Ceiling(R1) = 5 \geq \pi_{P2} = 5$. Process P4 can block P2.
(iii) Process P1 locks R1 and $Ceiling(R2) = 3 \ngeq \pi_{P2} = 5$. Process P1 cannot block P2.
(iv) P3 does not use resources hence it cannot block P2.

Process P2 can only be blocked by process P4. Then, the blocking time is

$B_{P2} = max(\, t_{R1,P4}) = 2ms.$

3) No, the schedulability test used for RMS cannot be applied because process P2 has a deadline D smaller than the period T.

Q1.4:

1) According to the description we have the following task set:

Process	Period	WCET
P1 – Servo motor controller 1	4ms	0.5ms
P2 – Servo motor controller 2	4ms	0.5ms
P3 – Sensor analysis	10ms	2ms
P4 – Decision maker	10ms	1ms
P5 – Logging	?	2ms

Since the scheduler used is EDF, we use the schedulability test formula to isolate the minimum possible period of the logging task:

$$\sum_{i=1}^{N} \left(\frac{C_i}{T_i}\right) \leq 1$$

$$\frac{C_{P1}}{T_{P1}} + \frac{C_{P2}}{T_{P2}} + \frac{C_{P3}}{T_{P3}} + \frac{C_{P4}}{T_{P4}} + \frac{C_{P5}}{T_{P5}} \leq 1$$

$$\frac{0.5}{4} + \frac{0.5}{4} + \frac{2}{10} + \frac{1}{10} + \frac{2}{T_{P5}} \leq 1$$

$$T_{P5} \geq 4.\bar{4}\, ms$$

The logging task could be run with a period of 4.5 ms or longer. Possible assumptions are:

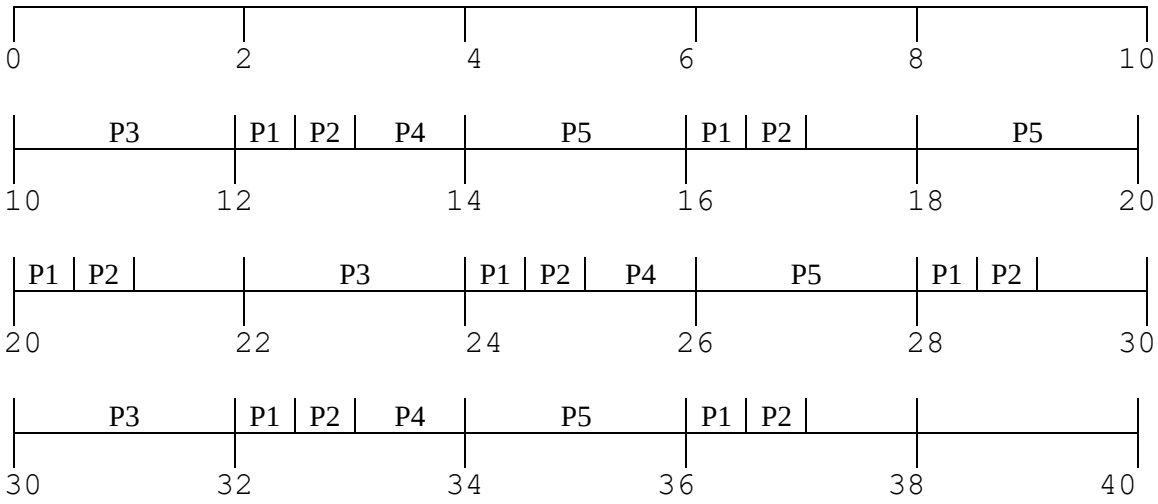
1.System overheads are not considered
2.Context switching is ignored
3. $D = T$

2) We calculate the major and minor cycles for the cyclic schedule:

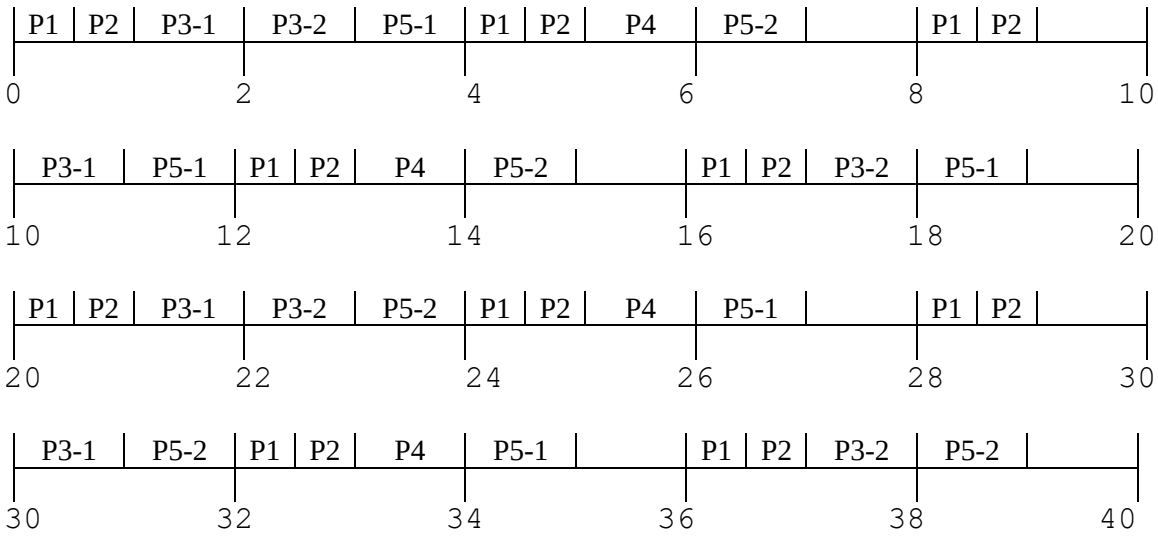
Minor cycle: gcd (4, 4, 10, 10, 8) = 2ms
Major cycle: lcm (4, 4, 10, 10, 8) = 40ms

A first attempt to a cyclic schedule is:

P1	P2		P3	P1	P2	P4	P5	P1	P2	



However, this schedule includes two times P3 within 10 ms (the deadline) from the first execution start (a similar problem exists for P5). In order to fix this, we assume that P3 and P5 can be split in two and design the following schedule:



Note: the above solution introduces jitter for P3-2.

3) As the processes with the same period are mixed this is the new task set:

Process	Period	WCET	Locking time M (shared memory)	Priority (π)
P1 – Servo motor controllers	4ms	1ms		3 (Highest)
P2 – Sensor and decision maker	10ms	3ms	0.5ms	1 (Lowest)
P3 – Logging	8ms	3ms	1ms	2

To calculate the response time analysis we have to take into account the blocking times. Then, first we have to calculate the ceiling of the resource M:

Ceiling (M) = max ($\pi 2$, $\pi 3$) = 2

The blocking times are calculated. In order to simplify the process it is recommended to go through the lower to higher priority processes. Let $lp\{Z\}$ denote the set of processes with lower priority than Z:

B_{P2}: There is no process with lower priority $lp\{P2\} = \{\}$. No process can block P2 then $B_{P2} = 0ms$.

B_{P3}: In this case $lp\{P3\} = \{P2\}$. We have to check if P3 can be blocked by the lower priority process P2:

(i) Process P2 locks M and $Ceiling(M) = 2 \geq \pi_{P3} = 2$. Process P2 can block P3.

Process P3 can be blocked by process P2. The blocking time is $B_{P3} = max(t_{M,P2}) = 0.5ms$.

B_{P1}: In this case $lp\{P1\} = \{P2, P3\}$. We have to check if P1 can be blocked by any of these two lower priority processes:

(i) Process P2 locks M and $Ceiling(M) = 2 \ngeq \pi_{P1} = 3$. Process P2 cannot block P1.
(ii) Process P3 locks M and $Ceiling(M) = 2 \ngeq \pi_{P1} = 3$. Process P3 cannot block P1.

Process P1 cannot be blocked by P2 neither P3. Then, the blocking time is $B_{P1} = 0ms$.

After computing the blocking time, we have all we need to do the response time analysis.

This time it is easier if we start with the highest priority task and we proceed down to the lowest.

R_{P1}: This is the process with highest priority, hence no other processes can interrupt it:

$$R_{P1} = C_{P1} + B_{P1} + J_{P1} = 1 + 0 + 0 = 1ms$$

R_{P3}: This process can be interrupted by P1.

$$w_{P3}^0 = C_{P3} + B_{P3} = 3 + 0.5 = 3.5ms$$
$$w_{P3}^1 = C_{P3} + B_{P3} + \left\lceil \frac{w_{P3}^0}{T_{P1}} \right\rceil C_{P1} = 3 + 0.5 + \left\lceil \frac{3.5}{4} \right\rceil 1 = 4.5ms$$
$$w_{P3}^2 = C_{P3} + B_{P3} + \left\lceil \frac{w_{P3}^1}{T_{P1}} \right\rceil C_{P1} = 3 + 0.5 + \left\lceil \frac{4.5}{4} \right\rceil 1 = 5.5ms$$
$$w_{P3}^3 = C_{P3} + B_{P3} + \left\lceil \frac{w_{P3}^2}{T_{P1}} \right\rceil C_{P1} = 3 + 0.5 + \left\lceil \frac{5.5}{4} \right\rceil 1 = 5.5ms$$

$$R_{P3} = w_{P3} + J_{P3} = 5.5 + 0 = 5.5ms$$

R_{P2}: This process can be interrupted by P1 and P3.

$$w_{P2}^{0} = C_{P2} + B_{P2} = 3 + 0 = 3ms$$

$$w_{P2}^{1} = C_{P2} + B_{P2} + \left\lceil \frac{w_{P2}^{0}}{T_{P1}} \right\rceil C_{P1} + \left\lceil \frac{w_{P2}^{0}}{T_{P3}} \right\rceil C_{P3} = 3 + 0 + \left\lceil \frac{3}{4} \right\rceil 1 + \left\lceil \frac{3}{8} \right\rceil 3 = 7ms$$

$$w_{P2}^{2} = C_{P2} + B_{P2} + \left\lceil \frac{w_{P2}^{1}}{T_{P1}} \right\rceil C_{P1} + \left\lceil \frac{w_{P2}^{1}}{T_{P3}} \right\rceil C_{P3} = 3 + 0 + \left\lceil \frac{7}{4} \right\rceil 1 + \left\lceil \frac{7}{8} \right\rceil 3 = 8ms$$

$$w_{P2}^{3} = C_{P2} + B_{P2} + \left\lceil \frac{w_{P2}^{2}}{T_{P1}} \right\rceil C_{P1} + \left\lceil \frac{w_{P2}^{2}}{T_{P3}} \right\rceil C_{P3} = 3 + 0 + \left\lceil \frac{8}{4} \right\rceil 1 + \left\lceil \frac{8}{8} \right\rceil 3 = 8ms$$

$$R_{P2} = w_{P2} + J_{P2} = 8 + 0 = 8ms$$

4) Sporadic task is a task that is not periodically scheduled, but for which we can set a minimum inter-arrival time. An example could be a task reacting to the external event produced by a user pressing the keys of a keyboard: here there is a minimum time we can consider before two consecutive events.

Q1.5:

1) From the description we obtain the following set of tasks:

Process	Period	WCET
P1 – Cruise controller	10ms	3ms
P2 – Wheel pair regulators	5ms	1ms
P3 – Automatic braking	?	1ms

In order to calculate the maximum possible period, we will create an equation with the formula that determines the CPU utilization of the task set and the maximum CPU utilisation allowed:

$$\sum_{i=1}^{N} \left(\frac{C_i}{T_i} \right) \leq 0.7$$

$$\frac{C_{P1}}{T_{P1}} + \frac{C_{P2}}{T_{P2}} + \frac{C_{P3}}{T_{P3}} \leq 0.7$$

$$\frac{3}{10} + \frac{1}{5} + \frac{1}{T_{P3}} \leq 0.7$$

$$T_{P3} \geq 5ms$$

The period of P3 must be equal or greater than 5ms.

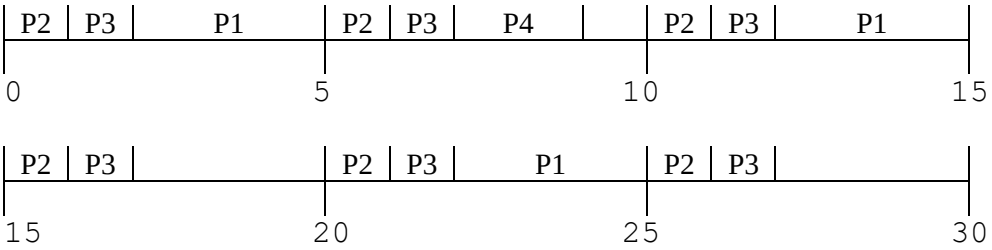
2) From the description we obtained the following set of tasks:

Process	Period	WCET
P1 – Cruise controller	10ms	3ms
P2 – Wheel pair regulators	5ms	1ms
P3 – Automatic braking	5ms	1ms
P4 – Power usage control	30ms	2ms

Then we calculate the major and minor cycles:

Minor cycle: gcd (10, 5, 5, 30) = 5ms
Major cycle: lcm (10, 5, 5, 30) = 30ms

And the resulting cycle schedule is:



3) The task set table that we can extract from the exercise description is the following:

Process	Priority (π)	Locking time R	Locking time S
P1	2	0.5ms	
P2	3	1ms	2ms
P3	1		1ms
P4	4		

The ceiling of the two resources are calculated:

Ceiling (R) = max (π1, π2) = 3
Ceiling (S) = max (π2, π3) = 3

The blocking times are calculated. In order to simplify the process it is recommended to go through the lower to higher priority processes. Let $lp\{Z\}$ denote the set of processes with lower priority than Z:

B_{P3}: There is no process with lower priority $lp\{P3\} = \{\}$. No process can block P3 then $B_{P3} = 0ms$.

B_{P1}: In this case $lp\{P1\} = \{P3\}$. We have to check if P1 can be blocked by the lower priority process P3:

(i) Process P3 locks S and $Ceiling(S) = 3 \geq \pi_{P1} = 2$. Process P3 can block P1.

Process P1 can be blocked by process P3. The blocking time is $B_{P1} = max(\, t_{S,P3}) = 1ms$.

B_{P2}: In this case $lp\{P2\} = \{P3, P1\}$. We have to check if P2 can be blocked by any of these two lower priority processes:

(i) Process P3 locks S and $Ceiling(S) = 3 \geq \pi_{P2} = 3$. Process P3 can block P2.
(ii) Process P1 locks R and $Ceiling(R) = 3 \geq \pi_{P2} = 3$. Process P1 can block P2.

Process P2 can be blocked by processes P3 and P1. Then, the blocking time is$B_{P2} = max(\, t_{S,P3}, t_{R,P1}) = 1ms$.

B_{P4}: In this case $lp\{P4\} = \{P3, P1, P2\}$. We have to check if P4 can be blocked by any of these lower priority processes:

(i) Process P3 locks S and $Ceiling(S) = 3 \ngeq \pi_{P4} = 4$. Process P3 cannot block P4.
(ii) Process P1 locks R and $Ceiling(R) = 3 \ngeq \pi_{P4} = 4$. Process P1 cannot block P4.
(iii) Process P2 locks R and S, and $Ceiling(R) = 3 \ngeq \pi_{P4} = 4$ and $Ceiling(S) = 3 \ngeq \pi_{P4} = 4$. Then process P2 cannot block P4.

Then $B_{P4} = 0ms$.

4) The statement is true because a task that is not periodic neither sporadic is aperiodic. An aperiodic task does not allow an execution time analysis because it does not present a minimum inter-arrival time.

Q1.6:

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
P3								**E**	**b3**	**b3**	**a3**	**a3**	**E**				
P1			**B**	**E**	**a3**	**E**	**b3**	**P**	**P**	**P**	**P**	**P**	**P**	**E**			
P2	**E**	**a3**	**a3**	**P**	**P**	**P**	**P**	**P**	**P**	**P**	**P**	**P**	**P**	**P**	**b3**	**b3**	**E**

Q1.7:

The priorities of the tasks, according to RMS, are:

Process	Priority (π)
A	2
B	4
C	3

D	1

a)
The ceiling of a semaphore is the maximum priority of all processes that may lock the semaphore. Denoting $\lceil X \rceil$ the ceiling of the resource X, we have that

$$\lceil S1 \rceil = max(\pi_A, \pi_B) = 4$$
$$\lceil S2 \rceil = max(\pi_C, \pi_D) = 3$$

b)
The blocking time for each process can be calculated with the following algorithm. Let $lp\{Z\}$ denote the set of processes with lower priority than Z.
It is convenient to start from the lowest priority process to the highest.
B_D: D is the lowest priority process, thus no other lower priority processes can block it. $lp\{D\} = \{\}$ thus$B_D = 0$.

B_A: $lp\{A\} = \{D\}$. Process D uses S2 and $\lceil S2 \rceil = 3 \geq \pi_A = 2$ then D can block process A since it can get a higher priority when blocking S2. The blocking time is $B_A = t_{S2,D} = 4$.

B_C: $lp\{C\} = \{D, A\}$. Let's examine which process can block C.
(i) Process D uses S2 and $\lceil S2 \rceil = 3 \geq \pi_C = 3$. Process D can block C.
(ii) Process A uses S1 and $\lceil S1 \rceil = 4 \geq \pi_C = 3$. Process A can block C.
Thus, process C is blocked at most once by the processes D and A. This means that $B_C = max(t_{S2,D}, t_{S1,A}) = 4$.

B_B: $lp\{B\} = \{D, A, C\}$.
(i) Process D uses S2 and $\lceil S2 \rceil = 3 \ngeq \pi_B = 4$. Process D cannot block B.
(ii) Process A uses S1 and $\lceil S1 \rceil = 4 \geq \pi_B = 4$. Process A can block B.
(iii) Process C uses S2 and $\lceil S2 \rceil = 3 \ngeq \pi_B = 4$. Process C cannot block B.
Only process A can block process B, then $B_B = t_{S1,A} = 1$.

Q1.8:
The priorities of the tasks, according to RMS, are:

Process	Priority (π)
A	1
B	2
C	3

Denoting $\lceil X \rceil$ the ceiling of the resource X, we have that:

$$\lceil S1 \rceil = max(\pi_A, \pi_B, \pi_C) = 3$$
$$\lceil S2 \rceil = max(\pi_B, \pi_C) = 3$$
$$\lceil S3 \rceil = max(\pi_C) = 3$$

As before, let $lp\{Z\}$ denote the set of processes with lower priority than Z.

B_A: A is the lowest priority of the set, thus $lp\{A\} = \{\}$. A cannot be blocked by any lower priority process, $B_A = 0$.

B_B: $lp\{B\} = \{A\}$. Process A uses S1 and $\lceil S1 \rceil = 3 \geq \pi_B = 2$ meaning that process A can block process B. The blocking time is$B_B = t_{S1,A} = 1$.

B_C: $lp\{C\} = \{A, B\}$.

(ii) Process A uses S1 and $\lceil S1 \rceil = 3 \geq \pi_C = 3$. Process A can block C.
(iii) Process B uses S1 and $\lceil S1 \rceil = 3 \geq \pi_C = 3$. Process B can block C when using the semaphore S1.
(iv) Process B uses S2 and $\lceil S2 \rceil = 3 \geq \pi_C = 3$. Process B can block C when using the semaphore S2.

Thus, process C is blocked at most once by the processes B and A. This means that $B_C = max(t_{S1,A}, t_{S1,B}, t_{S2,B}) = 1$.

We can now calculate the response time by solving the equation:

$$w_i = C_i + B_i + \sum_{\forall j \in hp(i)} \left\lceil \frac{w_i}{T_j} \right\rceil C_j \qquad R_i = w_i + J_i$$

Again, it is convenient to start from the highest priority process down to the lowest.
Response time R_C: hp(C)={} so the response time is not affected by interference from higher priority processes, then $R_C = C_1 + B_1 = 2 + 1 = 3$

Response time R_B: Only process C has higher priority than B, hp(B)={C}. Interference is caused by C and the response time is:

$$w_B^0 = C_B + B_B = 4$$
$$w_B^1 = C_B + B_B + \left\lceil \frac{w_B^0}{T_C} \right\rceil C_C = 6$$
$$w_B^2 = C_B + B_B + \left\lceil \frac{w_B^1}{T_C} \right\rceil C_C = 6$$
$$R_B = w_B^2 + J_B = 6 + 0 = 6$$

Response time R_A: Processes B and C have higher priority then A, hp(A)={B,C}. Interference has the contribution from B and C and the response time is:

$$w_A^0 = C_A + B_A = 4$$
$$w_A^1 = C_A + B_A + \left\lceil \frac{w_A^0}{T_B} \right\rceil C_B + \left\lceil \frac{w_A^0}{T_C} \right\rceil C_C = 9$$
$$w_A^2 = C_A + B_A + \left\lceil \frac{w_A^1}{T_B} \right\rceil C_B + \left\lceil \frac{w_A^1}{T_C} \right\rceil C_C = 11$$
$$w_A^3 = C_A + B_A + \left\lceil \frac{w_A^2}{T_B} \right\rceil C_B + \left\lceil \frac{w_A^2}{T_C} \right\rceil C_C = 14$$
$$w_A^4 = C_A + B_A + \left\lceil \frac{w_A^3}{T_B} \right\rceil C_B + \left\lceil \frac{w_A^3}{T_C} \right\rceil C_C = 14$$

$$R_A = w_A + J_B = 14 + 0 = 14$$