EXERCISE #2 – SERVICE SCHEDULING FEASIBILITY

REAL TIME EMBEDDED SYSTEMS (ECEN 5623)

SUMMER 2019

BOARD USED:

NVIDIA's JETSON NANO DEVELOPMENT BOARD

REPORT BY:

OM RAHEJA

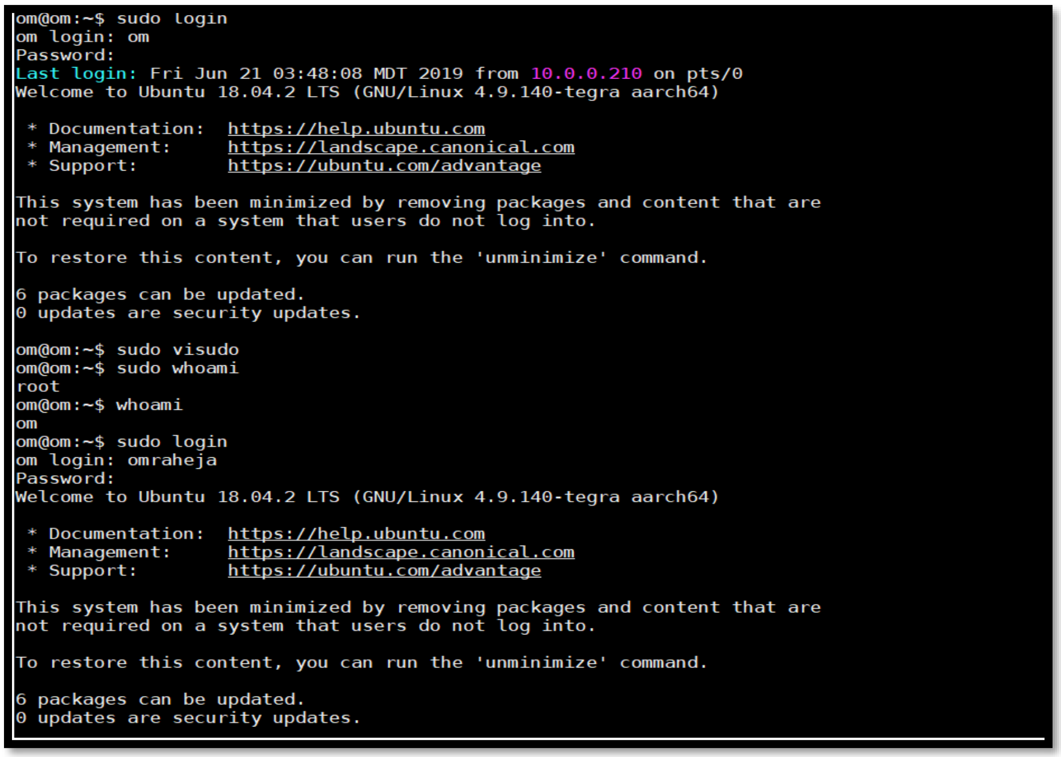
PROFESSOR:

DR. SAM SIEWERT

1. If you're using embedded Linux, make yourself an account on your R-Pi3b, Jetson Nano, or Jetson TK1 system. To do this on Jetson TKI, use the reset button if the system is locked, use the well-known "ubuntu" password to login, and then use "sudo adduser", enter the well-known password, and enter user information as you see fit. Add your new user account as a "sudoer" using "visudo" right below root with the same privileges (if you need help with "vi", here's a quick reference or reference card– use arrows to position cursor, below root hit Esc, "i" for insert, type username and privileges as above, and when done, Esc, ":", "wq"). The old unix vi editor was one of the first full-screen visual editors – it still has the advantage of being found on virtually any Unix system in existence but is otherwise cryptic – along with Emacs it is still widely used in IT, by developers and systems engineers, so it's good to know the basics. If you really don't like vi or Emacs, your next best bet is "nano" for Unix systems. Do a quick "sudo whoami" to demonstrate success. Logout of ubuntu and test your login, then logout. Use Alt+Print-Screen to capture your desktop and save as proof you set up your account. Note that you can always get a terminal with Ctrl+Alt+t key combination. If you don't like the desktop, you can try "GNOME Flashback" and please play around with customizing your account as you wish. Make sure you can access our class web page on Firefox (or default browser) and set your home page to http://ecee.colorado.edu/~ecen5623/index_summer.html . Overall, make sure you are comfortable with development, debug, compiler general native or cross-development tools and document and demonstrate that you know them.

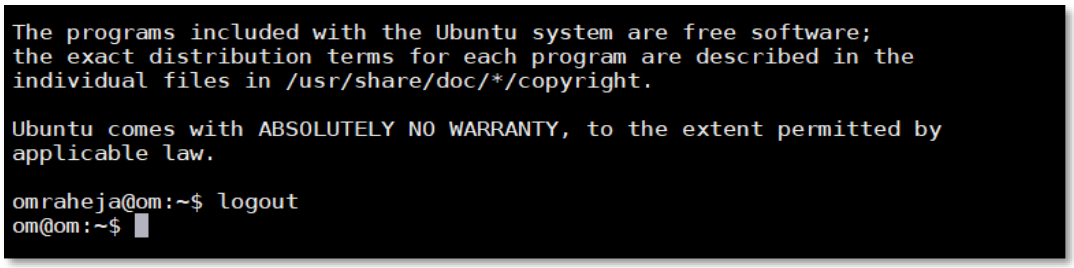
Ans:

CREATE ACCOUNT ON JETSON



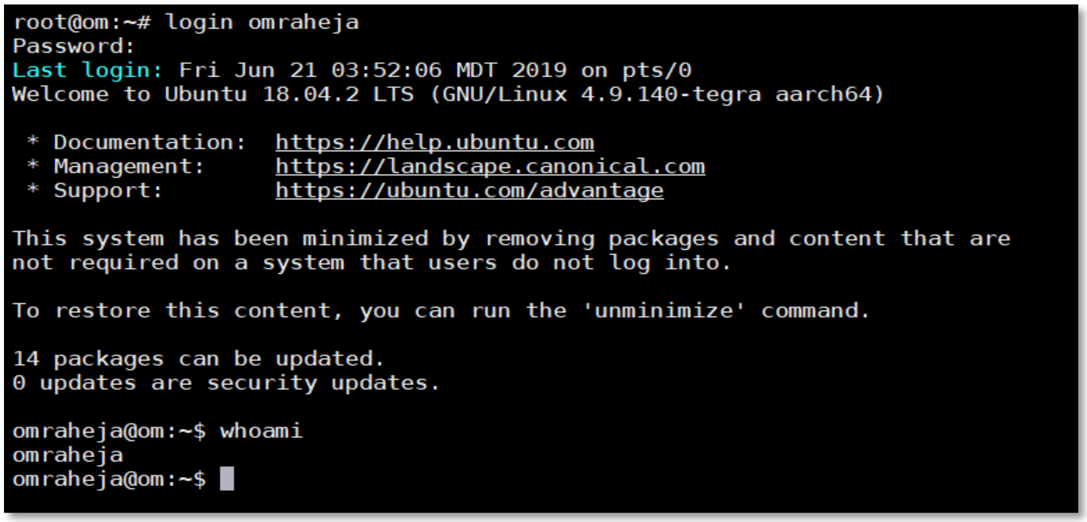
```
om@om:~$ sudo login
om login: om
Password:
Last login: Fri Jun 21 03:48:08 MDT 2019 from 10.0.0.210 on pts/0
Welcome to Ubuntu 18.04.2 LTS (GNU/Linux 4.9.140-tegra aarch64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.

6 packages can be updated.
0 updates are security updates.

om@om:~$ sudo visudo
om@om:~$ sudo whoami
root
om@om:~$ whoami
om
om@om:~$ sudo login
om login: omraheja
Password:
Welcome to Ubuntu 18.04.2 LTS (GNU/Linux 4.9.140-tegra aarch64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.

6 packages can be updated.
0 updates are security updates.
```

Figure 1

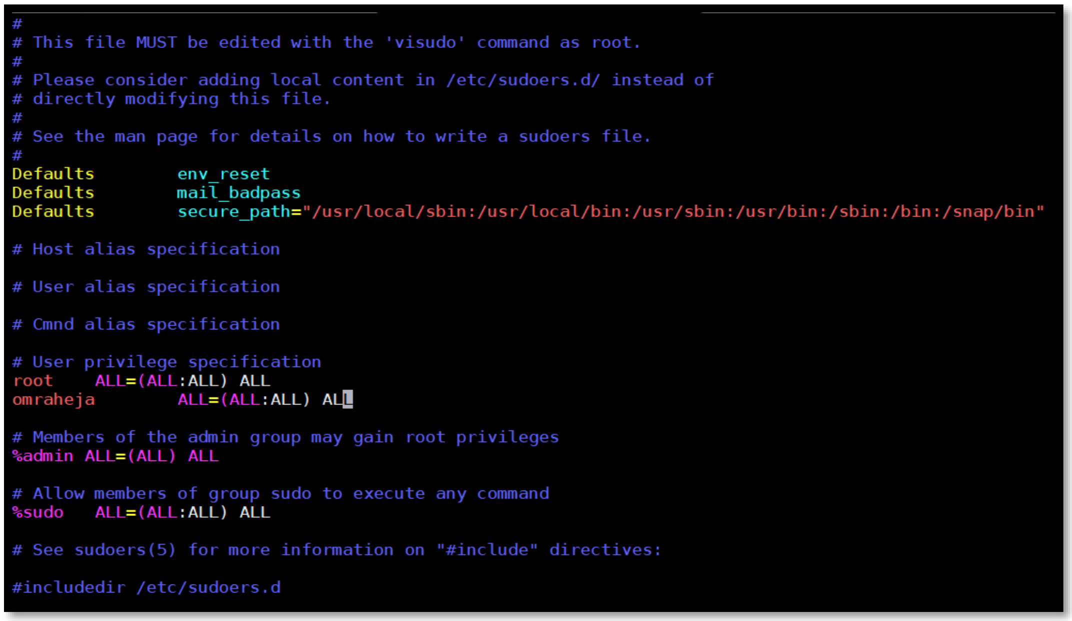


```
The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

omraheja@om:~$ logout
om@om:~$
```

Figure 2



```
root@om:~# login omraheja
Password:
Last login: Fri Jun 21 03:52:06 MDT 2019 on pts/0
Welcome to Ubuntu 18.04.2 LTS (GNU/Linux 4.9.140-tegra aarch64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.

14 packages can be updated.
0 updates are security updates.

omraheja@om:~$ whoami
omraheja
omraheja@om:~$
```



```
#
# This file MUST be edited with the 'visudo' command as root.
#
# Please consider adding local content in /etc/sudoers.d/ instead of
# directly modifying this file.
#
# See the man page for details on how to write a sudoers file.
#
Defaults        env_reset
Defaults        mail_badpass
Defaults        secure_path="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/snap/bin"

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root    ALL=(ALL:ALL) ALL
omraheja        ALL=(ALL:ALL) ALL

# Members of the admin group may gain root privileges
%admin ALL=(ALL) ALL

# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL

# See sudoers(5) for more information on "#include" directives:

#includedir /etc/sudoers.d
```

Figure 3

2. Read the paper "Architecture of the Space Shuttle Primary Avionics Software System" [available on Canvas], by Gene Carlow and provide an explanation and critique of the frequency executive architecture. What advantages and disadvantages does the frequency executive have compared to the real-time threading and tasking implementation methods for real-time software systems? Please be specific about the advantages and disadvantages and provide at least 3 advantages as well as 3 disadvantages.

Ans:

SHUTTLE PAPER REVIEW

Overall understanding of paper and key point articulation:

The paper describes about the Primary Avionics Software System (PASS). PASS is considered the most complex flight computer program ever developed. It epitomizes the benefits to be gained by establishing a well-structured system architecture.

The introductory paragraph says that by using a structured approach to programming techniques can lead to improved software quality and maintainability. Following this approach has led to compilers being more reliable and efficient. This even led to improved control and stability of the development process. All these mentioned attributes would not be that efficient if the overall architecture has not been well-structured.

PASS performs a critical role in virtually all operational aspects of the Space shuttle orbiter. The system consists of multiple computer redundancy modules to support the system in case of any fault.

PASS ARCHITECTURAL DRIVERS

PASS is central to all the Space shuttle orbiter avionics system functions which include vehicle flight control. There are many factors in addition to the critical timing and redundancy management requirements of the avionic system design that have influenced the architecture of the PASS. Data processing system (DPS)/ IBM AP-101 GPC (general purpose computer) design and memory/CPU constraints; the multi-computer redundancy management and synchronization; the operational sequencing/mode control and man/machine interface requirements; the applications functional and performance requirements; and the requirement for design modularity and modification flexibility are a few factors that have been a vital part of the PASS. When these modules are successfully accommodated, it results in the development of a sound software system architecture.

OPERATIONAL STRUCTURE

The operational structure consists of the main memory with a capacity of AP-101 (106K 32-bit words). However, this memory is not sufficient to contain all the software (500K) which is required to satisfy all the requirements of the PASS. The mass memory that was used in the PASS was serial device and not a direct-access storage device. The reliability considerations dictate that the software required to support a critical mission phase (such as ascent) must be resident in each of the redundant GPC main memory throughout the phase. This is done in order to avoid total system failure. The purpose of using redundant modules in a system is to ensure that the system continues to work if the main module fails in any case. This way, tasks performed by the main module is switched over to the redundant modules. Thus, it is very important to feed identical data to the redundant modules as present on the main system module, so that in case the main system fails, the redundant modules have all the resources and data to carry forward the system from there.

PASS has been a segmentation of eight different phase/function combinations, each of which is identified by a unique OPS (operational sequence) designation. The software that is required for each OPS is loaded into the GPC main memory from the mass memory. This is done during initiation of the operational sequence. A few non-critical situations like loading of the display during on-orbit coast periods are not transferred to the GPC's main memory during the execution of an OPS.

PASS is stored in the main memory in a manner such that it minimizes the amount of code and data that must be transferred to the main memory when a new OPS is initiated. The main memory load of each OPS has three parts:

- Resident or system software – contains code and data common to all OPS loads
- Major function base – contains code and data common to major applications functions used in more than more OPS load
- OPS overlay – contains the applications code and data unique to an OPS load.

Loading in main memory occurs during the transition from one OPS to another in response to a major application function. This switch in the function can be an entry from the crew. The contents of the OPS that is currently in progress determine which out of the three parts must be loaded for support of the new OPS. The best case would be that only the OPS overlay has to be loaded in the main memory.

MAN/MACHINE INTERFACE

To have a user interface in any system in considered advantageous because it helps a knowledgeable user to keep track of the overall system. The case for the PASS system was no different. It consisted of a man-machine interface such that the crew could communicate with the computers to monitor and control the orbiter avionic system. The man-machine interface was implemented as a substructure of the OPS.

Each OPS has more than one mode. Major modes are sub-structured into blocks linked to CRT displays. Switching from one mode/block to another occurred either when a keyboard entry was made or in some cases, automatically as a result of a specific event or condition detected by the software. At any given time, at least one major mode will be entered when the OPS is initiated.

The second element of the OPS sub structure is SPEC (specialist function). SPEC is initiated when a keyboard entry is made by the crew. After initiation, the SPEC executes independently and concurrently with other processes that take place within the OPS. This substructure is also connected to the CRT displays which help the crew to monitor the processing results.

Third element of OPS substructure: DISP (display function). This does not require any initialization.

Role of major modes: Accomplish primary functions in OPS

Role of SPEC: Secondary or background functions.

Role of DISP: Used to display the processing results in the system.

Control segment facilitates the implementation of the OPS. The control segments consist of a series of standard logic blocks that establish structure of OS, major mode, SPEC or DISP. The control segment grammar consists of a library of macros to standardize and modularize the design and implementation of control segments.

SYSTEM SOFTWARE

The software architecture adopted was a synchronous design approach within which each application process was dispatched in a timed manner so that it occurs at a specific point relative to the start of the system cycle. To prevent any overruns at the system and process levels, care must be taken to synchronize the start and end of associated IO.

However, as the shuttle computers performed a number and variety of tasks, they decided to use a nonsynchronous approach for the PASS architecture.

FLIGHT COMPUTER OPERATING SYSTEM (FCOS)

FCOS managed and controlled the internal resources of the GPC and other external interfaces.

System control task:

- Loading and Initialization.
- Inter computer communication & redundant computer synchronization.

User Interface task:

- Communications between user and system.

Three major tasks performed by FCOS:

Process management:

- Controls allocation of all internal computer resources in response to requests from other systems.
- Schedules and allocates CPU resources based on priority queue structure.
- Ensures that highest priority system or process which is ready gets the CPU resources.

IO management:

- Controls allocation of IO processor (IOP) resources.
- Encapsulates the IOP software.

DPS configuration management:

- Load GPC memories and sequencing.
- Control of GPC and IOP operating states, including hardware initialization and status checks.
- Transfer of code from mass memory to GPC main memory.
- Program overlay and inflight mass memory modification.

SYSTEM CONTROL

Tasks performed by the system control are:

- Initialization and configuration control of DPS and associated avionic data network.
- Establish an interface and primary/secondary relationship with other GPCs.
- System level SPECS perform functions like resetting MTU, initiate memory dump, examine and initiate core location in memory, change configuration of DPS.

APPLICATIONS SOFTWARE

This software was developed for three major applications: Guidance, Navigation and Control (GN&C), Vehicle system management (SM) and Vehicle checkout (VCO).

Guidance, Navigation and Control (GN&C): It's a cyclic closed loop application. Performs a wide variety of time critical tasks. Determines vehicle position, velocity and altitude; Performs sensor redundancy management. Provides crew with displays and data entry capabilities to monitor. Critical flight control processing is completed within 40 millisecond minor cycles. The principal and vital functions within an OPS executed in the frequency range of 25Hz to 0.25Hz depending upon the application. The higher frequency executive was scheduled at high priority at a rate of 25Hz. On the other hand, lower critical tasks were scheduled at medium or low frequency.

Vehicle system management (SM): Monitors system performance and configuration of orbiter. Monitors payload subsystems to detect any abnormal conditions and alert the crew about it. The data acquisition is executed at a rate of 5Hz by the system management OPS control segment.

Vehicle checkout (VCO): Provides software support for testing, integration or certification of orbiter avionics subsystems during vehicle preparation.

Three advantages of frequency executive:

1) As in frequency executive method, preemption between tasks is not supported, the overhead of context switching is none. In case of real time thread scheduling, higher priority tasks can preempt the lower priority tasks. This requires a context switch, meaning time will be wasted in saving the present state of the systems, and storing the values in various registers.
2) The tasks in frequency executive method run in a cyclic period hyperloop. This means that the schedule is predictable. Because of this reason, the system can be said to be more deterministic.
3) As the schedule is predictable and repeatable, it is much easier to implement as compared to the real time threading implementation.

Three disadvantages of frequency executive:

1) This approach has limited flexibility to accommodate change. If parameters like priority and deadline for a service are changed, then this will have an impact on other services as well. On the other hand, in real-time embedded systems, it is easy to change parameters for one thread without affecting the other.
2) Task scheduling in frequency executive approach is non-preemptive. Meaning, if a lower priority task in activated as a result of some event then a task with higher priority cannot preempt it

unless the lower priority task runs to completion. On the other hand, in real time operating systems, scheduling is done in a manner which supports preemption of lower priority tasks. This way, the higher priority task does not have to wait for the lower priority task to finish its completion.

3) In frequency executive method, all the tasks that could execute together must be analyzed together. In real time thread scheduling each thread is assigned a fixed/dynamic scheduling priority according to which they execute.
4) If the number and variety of applications increase for a system, then it becomes difficult to schedule and synchronize the tasks in case of frequency executive method. Complexity for real time threading implementation is comparatively less than the later method.

3. Download feasibility example code and build it on a Jetson or alternate system of your choice (or ECES Linux if you have not mastered the Jetson yet) and execute the code. Compare the tests provided to analysis using Cheddar for the first 4 examples tested by the code. Now, implement remaining examples [5 more] of your interest from those that we reviewed in class (found here). Complete analysis using Cheddar RM. In cases where RM fails, test EDF or LLF to see if it succeeds and if it does, explain why. Cheddar uses both service simulations over the LCM of the periods as well as feasibility analysis based on the RM LUB and scheduling-point/completion-test algorithms, referred to as "Worst Case Analysis". Does your modified Feasibility code agree with Cheddar analysis in all 5 additional cases? Why or why not?

Ans: <u>Test Results for Feasibility Example Code:</u>

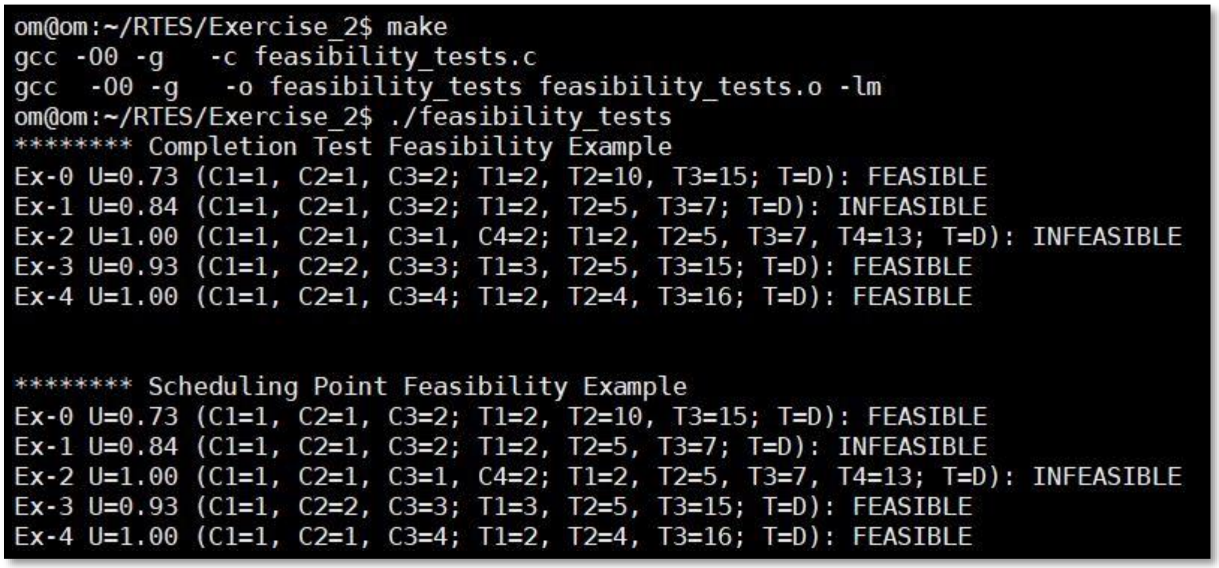

```
om@om:~/RTES/Exercise_2$ make
gcc -O0 -g   -c feasibility_tests.c
gcc  -O0 -g   -o feasibility_tests feasibility_tests.o -lm
om@om:~/RTES/Exercise_2$ ./feasibility_tests
******** Completion Test Feasibility Example
Ex-0 U=0.73 (C1=1, C2=1, C3=2; T1=2, T2=10, T3=15; T=D): FEASIBLE
Ex-1 U=0.84 (C1=1, C2=1, C3=2; T1=2, T2=5, T3=7; T=D): INFEASIBLE
Ex-2 U=1.00 (C1=1, C2=1, C3=1, C4=2; T1=2, T2=5, T3=7, T4=13; T=D): INFEASIBLE
Ex-3 U=0.93 (C1=1, C2=2, C3=3; T1=3, T2=5, T3=15; T=D): FEASIBLE
Ex-4 U=1.00 (C1=1, C2=1, C3=4; T1=2, T2=4, T3=16; T=D): FEASIBLE


******** Scheduling Point Feasibility Example
Ex-0 U=0.73 (C1=1, C2=1, C3=2; T1=2, T2=10, T3=15; T=D): FEASIBLE
Ex-1 U=0.84 (C1=1, C2=1, C3=2; T1=2, T2=5, T3=7; T=D): INFEASIBLE
Ex-2 U=1.00 (C1=1, C2=1, C3=1, C4=2; T1=2, T2=5, T3=7, T4=13; T=D): INFEASIBLE
Ex-3 U=0.93 (C1=1, C2=2, C3=3; T1=3, T2=5, T3=15; T=D): FEASIBLE
Ex-4 U=1.00 (C1=1, C2=1, C3=4; T1=2, T2=4, T3=16; T=D): FEASIBLE
```

Figure 4

<u>Overview of the code:</u>

In the feasibility example code provided, it checks the feasibility of 5 service sets by stressing them under completion test and scheduling point test. If the service sets pass these two tests, then they are said to be feasible.

<u>Detailed explanation of the code:</u>

The code starts is execution from the main function. At the beginning of analysis of each service set, its CPU utilization is carried out. This is done by using the formula U = ∑ (C_i/T_i), where U denotes CPU Utilization, C_i denotes run-time of each task in the service set, T_i denotes the time period of each task in the service set and 'i' varies from 1 to m(number of tasks). Once, the CPU utilization for a service set has been carried out, its value is printed.

After calculating and displaying the CPU utilization, the number of tasks/services have been calculated. This is passed as a parameter to the completion_time_feasibility function and scheduling_point_feasibility function.

The completion time feasibility test and the scheduling point feasibility test are the carried out. The parameters required by the two above mentioned functions are

a) Number of services (numServices)
b) ex?_period
c) ex?_wcet
d) ex?_deadline

where, **?** denotes the service number for which these parameters are defined. For example, for service number 2, the parameters will be denoted as numServices, ex2_period, ex2_wcet, ex2_deadline.

In most of the cases, according to RM policy, the deadline of each task is same as its time period. That means, the service/task has to run to completion before it invoked next.

Completion Time Test:

The completion time test serves as an alternative to the scheduling point test. The completion time test says that,

$\mathbf{A_n(t) = \sum_{j=1} (\lceil t/T_j \rceil)* C_j}$

Where,

$\lceil t/T_j \rceil$ denotes the number of executions of S_j at time t.

$(\lceil t/T_j \rceil)* C_j$ denotes the demand of S_j in time at t.

$A_n(t)$ is the total cumulative demand from the n tasks up to time t.

If we are able to prove that $A_n(t)$ is less than or equal to the deadline for S_n, then it proves that S_n is feasible.

The completion_time_feasibility function tests whether all services in the set mentioned are feasible or not. If the service set is feasible for all the services in the set then the set is said to be feasible. If the sets miss their deadline then it is said that the service set in infeasible according to the RM policy.

The function calculates the value of A_n for each service and determines if its less than or equal to the deadline for that task. Depending upon this condition, it is decided whether the service set passes the completion time test or not.

Scheduling Point Test:

According to Lehoczky, Sha, and Ding theorem, if a set of services can be shown to meet all deadlines from the critical instant up to the longest deadline of all tasks in the set, then the set is feasible. Assumption made by Liu Layland was that in worst case, all services might be requested at the same point in time. Based on this assumption, Lehoczky, Sha and Ding proposed a scheduling point test to determine whether or not a set of services are feasible.

The theorem states that,

$$\forall i, 1 \le i \le n, \min \sum_{j=1}^{i} C_j \left\lceil \frac{(l)T_k}{T_j} \right\rceil \le (l)T_k$$
$$(k,l) \in R_i$$
$$R_i = \left\{ (k,l) \mid 1 \le k \le i, l = 1, \ldots, \left\lfloor \frac{T_i}{T_k} \right\rfloor \right\}$$

- Where n is the number of tasks in the set S_i to S_n, where S_1 has higher priority than S_2, and S_n has higher priority than $S_{n>1}$.
- j identifies S_j, a service in the set between S_1 and S_n.
- k identifies S_k, a service whose l periods must be analyzed.
- l represents the number of periods of S_k to be analyzed.
- $\left\lceil \frac{(l)T_k}{T_j} \right\rceil$ represents the number of times S_j executes within l period of S_k.

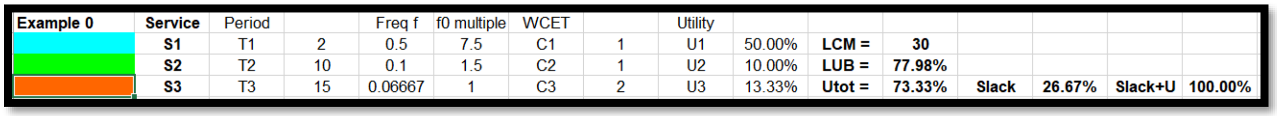
$C_j \left\lceil \frac{(l)T_k}{T_j} \right\rceil$ is the time required by S_j to execute within 1 periods of S_k. If the sum of these times for the set of tasks is smaller than 1 period of S_k, then the service set is feasible. The code given in the feasibility test example calculates the abovementioned parameters and checks the condition to determine if that particular task set is feasible or not.

CHEDDAR ANALYSIS

SERVICE SET 0

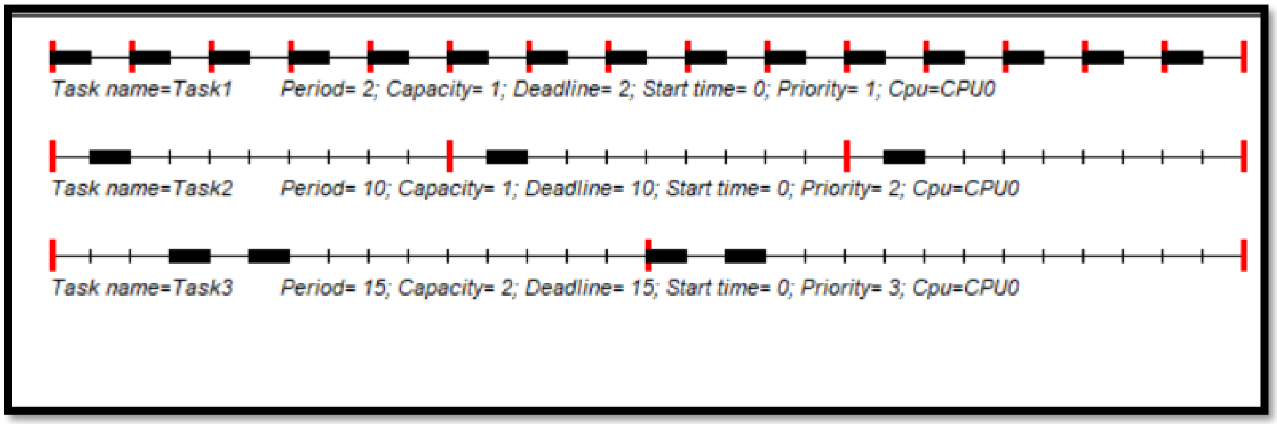
Time periods: T1=2, T2=10, T3=15

Run-Times: C1=1, C2=1, C3=2



Example 0	Service	Period		Freq f	f0 multiple	WCET		Utility							
🟦	S1	T1	2	0.5	7.5	C1	1	U1	50.00%	LCM =	30				
🟩	S2	T2	10	0.1	1.5	C2	1	U2	10.00%	LUB =	77.98%				
🟧	S3	T3	15	0.06667	1	C3	2	U3	13.33%	Utot =	73.33%	Slack	26.67%	Slack+U	100.00%

RATE MONOTONIC ANALYSIS:

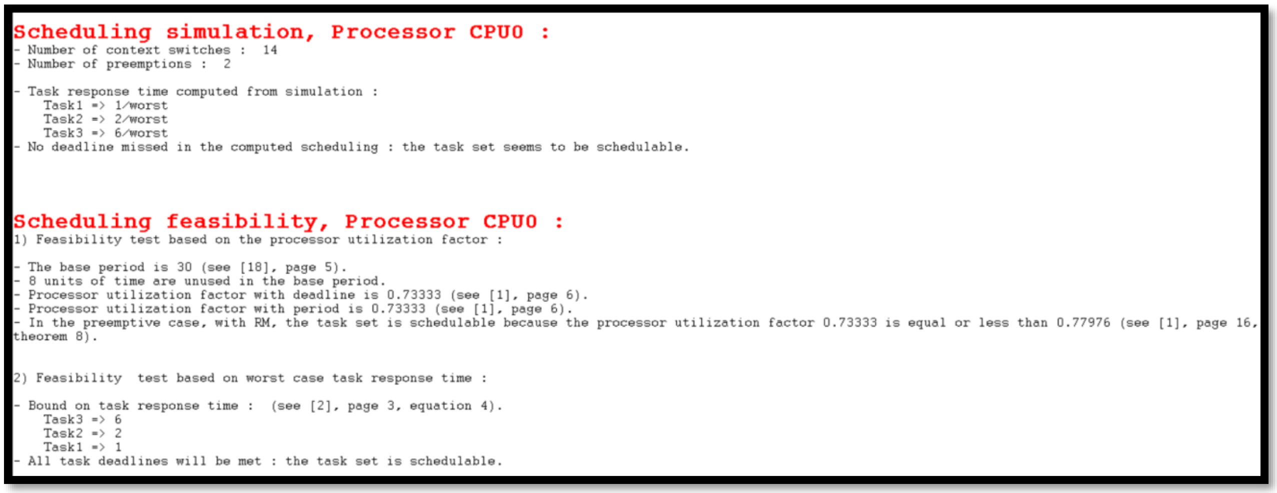


Task name=Task1 Period= 2; Capacity= 1; Deadline= 2; Start time= 0; Priority= 1; Cpu=CPU0

Task name=Task2 Period= 10; Capacity= 1; Deadline= 10; Start time= 0; Priority= 2; Cpu=CPU0

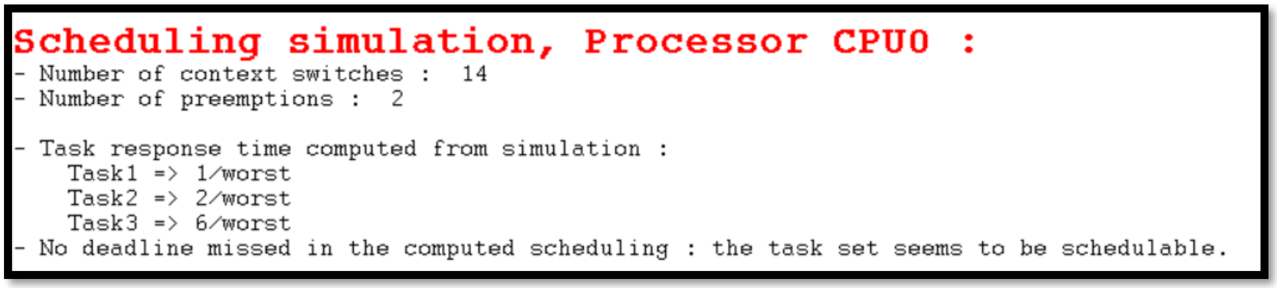
Task name=Task3 Period= 15; Capacity= 2; Deadline= 15; Start time= 0; Priority= 3; Cpu=CPU0

Figure 5



```
Scheduling simulation, Processor CPU0 :
- Number of context switches :  14
- Number of preemptions :  2

- Task response time computed from simulation :
    Task1 => 1/worst
    Task2 => 2/worst
    Task3 => 6/worst
- No deadline missed in the computed scheduling : the task set seems to be schedulable.


Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The base period is 30 (see [18], page 5).
- 8 units of time are unused in the base period.
- Processor utilization factor with deadline is 0.73333 (see [1], page 6).
- Processor utilization factor with period is 0.73333 (see [1], page 6).
- In the preemptive case, with RM, the task set is schedulable because the processor utilization factor 0.73333 is equal or less than 0.77976 (see [1], page 16,
theorem 8).


2) Feasibility  test based on worst case task response time :

- Bound on task response time :  (see [2], page 3, equation 4).
    Task3 => 6
    Task2 => 2
    Task1 => 1
- All task deadlines will be met : the task set is schedulable.
```

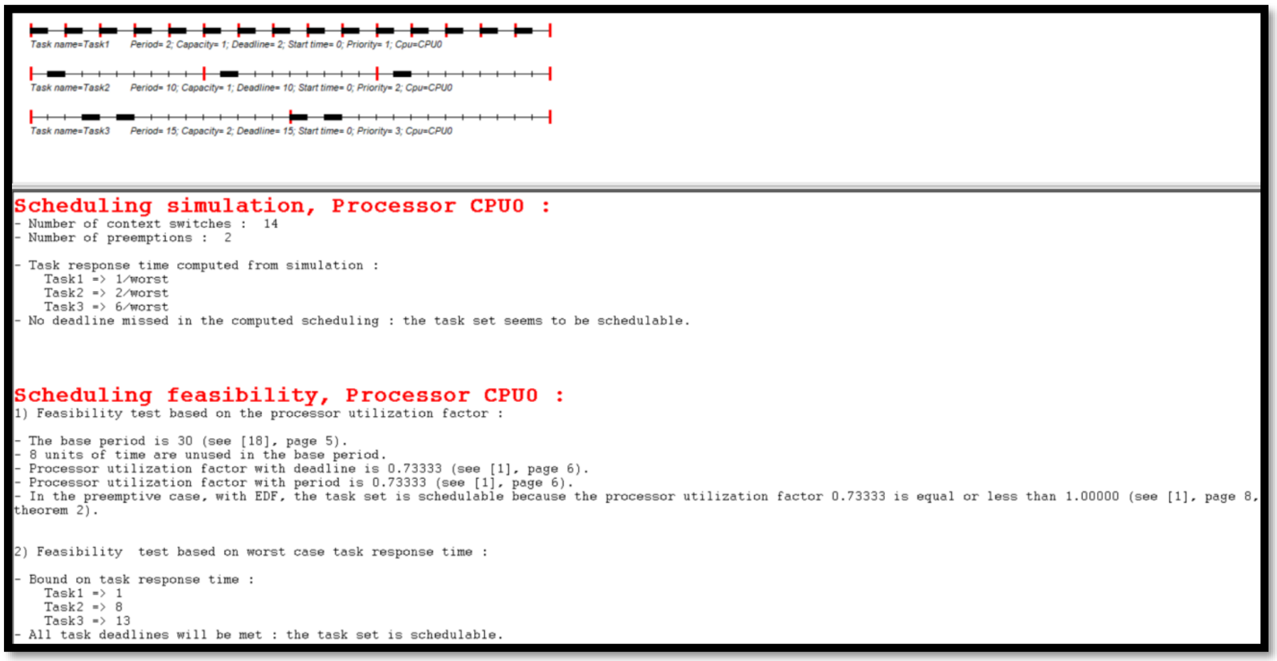
Figure 6



```
Scheduling simulation, Processor CPU0 :
- Number of context switches :  14
- Number of preemptions :  2

- Task response time computed from simulation :
    Task1 => 1/worst
    Task2 => 2/worst
    Task3 => 6/worst
- No deadline missed in the computed scheduling : the task set seems to be schedulable.
```

Figure 7

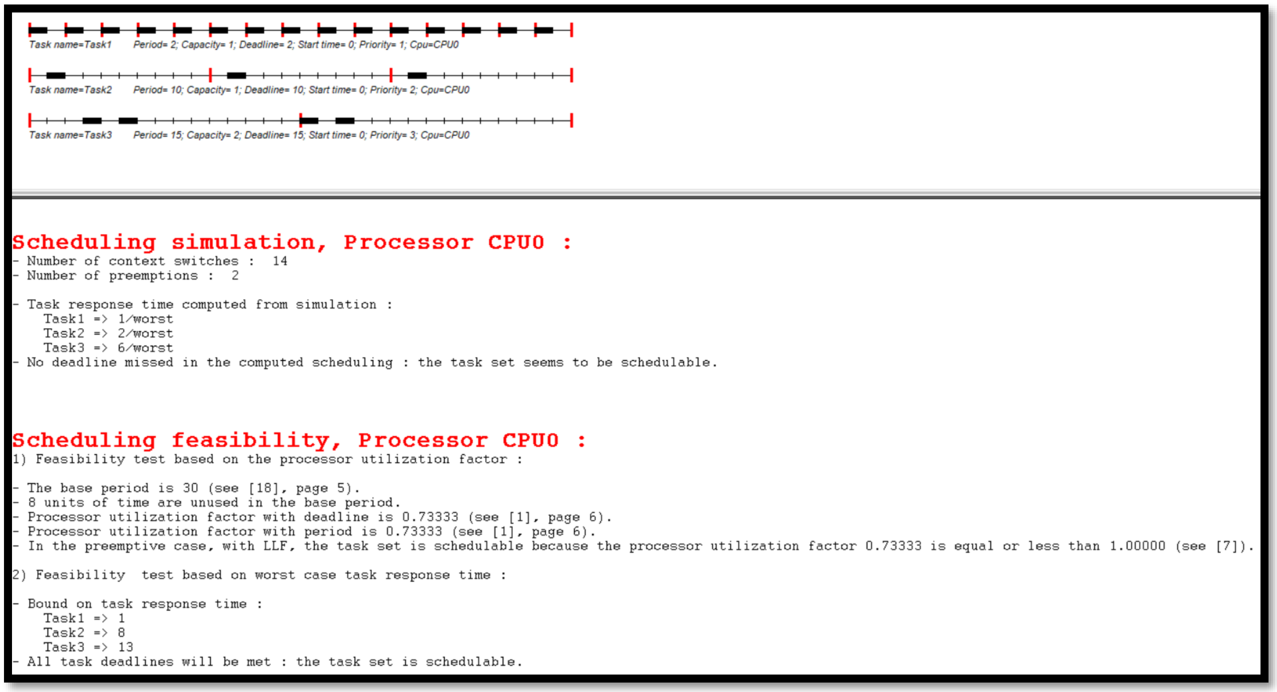
EARLIEST DEADLINE FIRST ANALYSIS:


Task name=Task1 Period= 2; Capacity= 1; Deadline= 2; Start time= 0; Priority= 1; Cpu=CPU0
Task name=Task2 Period= 10; Capacity= 1; Deadline= 10; Start time= 0; Priority= 2; Cpu=CPU0
Task name=Task3 Period= 15; Capacity= 2; Deadline= 15; Start time= 0; Priority= 3; Cpu=CPU0

```
Scheduling simulation, Processor CPU0 :
- Number of context switches :  14
- Number of preemptions :  2

- Task response time computed from simulation :
    Task1 => 1/worst
    Task2 => 2/worst
    Task3 => 6/worst
- No deadline missed in the computed scheduling : the task set seems to be schedulable.


Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The base period is 30 (see [18], page 5).
- 8 units of time are unused in the base period.
- Processor utilization factor with deadline is 0.73333 (see [1], page 6).
- Processor utilization factor with period is 0.73333 (see [1], page 6).
- In the preemptive case, with EDF, the task set is schedulable because the processor utilization factor 0.73333 is equal or less than 1.00000 (see [1], page 8,
theorem 2).


2) Feasibility  test based on worst case task response time :

- Bound on task response time :
    Task1 => 1
    Task2 => 8
    Task3 => 13
- All task deadlines will be met : the task set is schedulable.
```

Figure 8

<u>LEAST LAXITY FIRST ANALYSIS:</u>


Task name=Task1 Period= 2; Capacity= 1; Deadline= 2; Start time= 0; Priority= 1; Cpu=CPU0
Task name=Task2 Period= 10; Capacity= 1; Deadline= 10; Start time= 0; Priority= 2; Cpu=CPU0
Task name=Task3 Period= 15; Capacity= 2; Deadline= 15; Start time= 0; Priority= 3; Cpu=CPU0

```
Scheduling simulation, Processor CPU0 :
- Number of context switches :  14
- Number of preemptions :  2

- Task response time computed from simulation :
    Task1 => 1/worst
    Task2 => 2/worst
    Task3 => 6/worst
- No deadline missed in the computed scheduling : the task set seems to be schedulable.


Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The base period is 30 (see [18], page 5).
- 8 units of time are unused in the base period.
- Processor utilization factor with deadline is 0.73333 (see [1], page 6).
- Processor utilization factor with period is 0.73333 (see [1], page 6).
- In the preemptive case, with LLF, the task set is schedulable because the processor utilization factor 0.73333 is equal or less than 1.00000 (see [7]).

2) Feasibility  test based on worst case task response time :

- Bound on task response time :
    Task1 => 1
    Task2 => 8
    Task3 => 13
- All task deadlines will be met : the task set is schedulable.
```

Figure 9

<u>EXPLANATION FOR SERVICE SET 0</u>: From figure 5,6,7,8,9 and 32, it can be seen that the service set can be scheduled. The output of the code mentions that the service set is Feasible meaning, it has successfully passed the completion time test and the scheduling point test. This was confirmed by cheddar analysis for that service set. The rate monotonic test depicts that in a preemptive case, with RM, the task set is schedulable because the processor utilization factor is less than or equal to 1.00. It can also be seen that the task set is also schedulable with earliest deadline first and least laxity first.

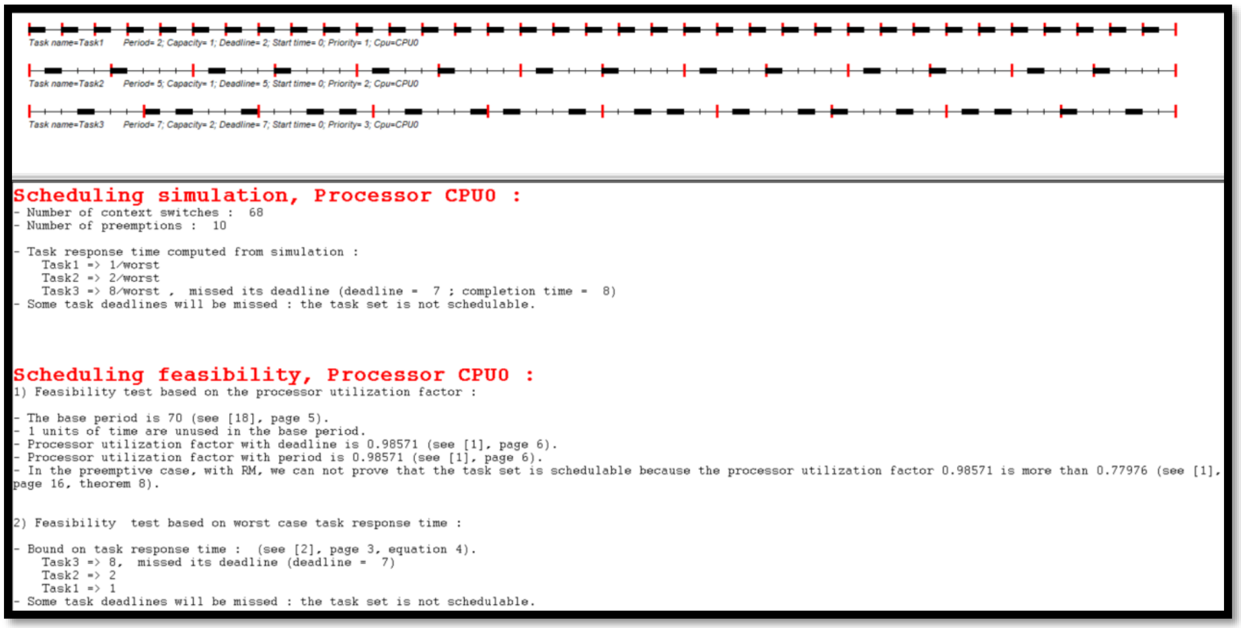
SERVICE SET 1

Time Periods: T1=2 T2=5 T3=7

Run-Times: C1=1 C2=1 C3=2

Example 1	Service	Period		Freq f	f0 multiple	WCET		Utility			
🟦	S1	T1	2	0.5	3.5	C1	1	U1	50.00%	**LCM =**	**70**
🟩	S2	T2	5	0.2	1.4	C2	1	U2	20.00%	**LUB =**	**77.98%**
🟧	S3	T3	7	0.142857	1	C3	2	U3	28.57%	**Utot =**	**98.57%**

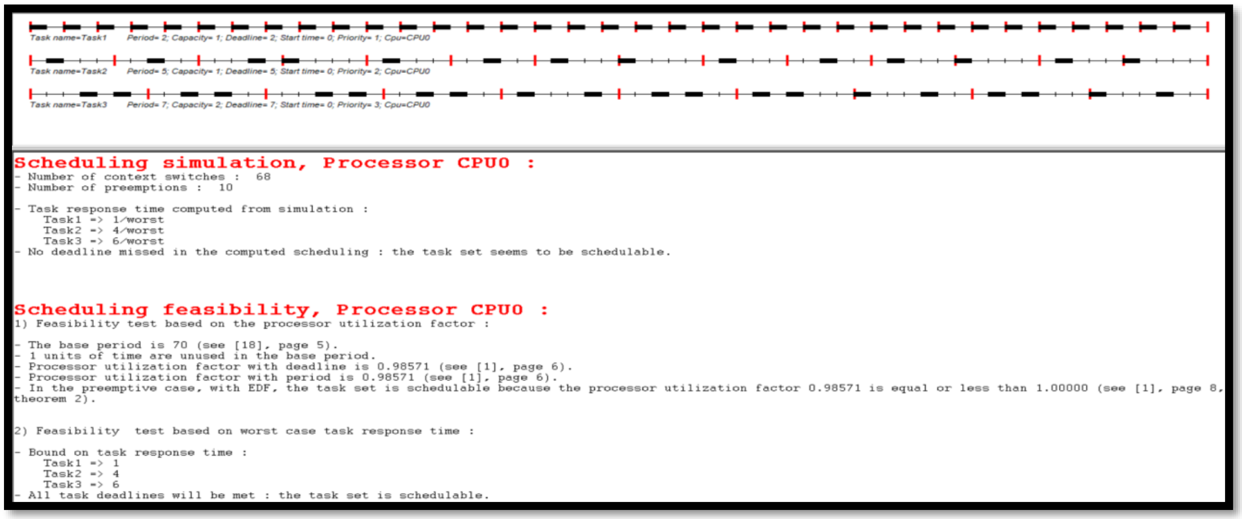
RATE MONOTONIC ANALYSIS:


Task name=Task1 Period= 2; Capacity= 1; Deadline= 2; Start time= 0; Priority= 1; Cpu=CPU0
Task name=Task2 Period= 5; Capacity= 1; Deadline= 5; Start time= 0; Priority= 2; Cpu=CPU0
Task name=Task3 Period= 7; Capacity= 2; Deadline= 7; Start time= 0; Priority= 3; Cpu=CPU0

```
Scheduling simulation, Processor CPU0 :
- Number of context switches :  68
- Number of preemptions :  10

- Task response time computed from simulation :
    Task1 => 1/worst
    Task2 => 2/worst
    Task3 => 8/worst ,  missed its deadline (deadline =  7 ; completion time =  8)
- Some task deadlines will be missed : the task set is not schedulable.



Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The base period is 70 (see [18], page 5).
- 1 units of time are unused in the base period.
- Processor utilization factor with deadline is 0.98571 (see [1], page 6).
- Processor utilization factor with period is 0.98571 (see [1], page 6).
- In the preemptive case, with RM, we can not prove that the task set is schedulable because the processor utilization factor 0.98571 is more than 0.77976 (see [1],
page 16, theorem 8).


2) Feasibility  test based on worst case task response time :

- Bound on task response time :  (see [2], page 3, equation 4).
    Task3 => 8,  missed its deadline (deadline =  7)
    Task2 => 2
    Task1 => 1
- Some task deadlines will be missed : the task set is not schedulable.
```

Figure 10

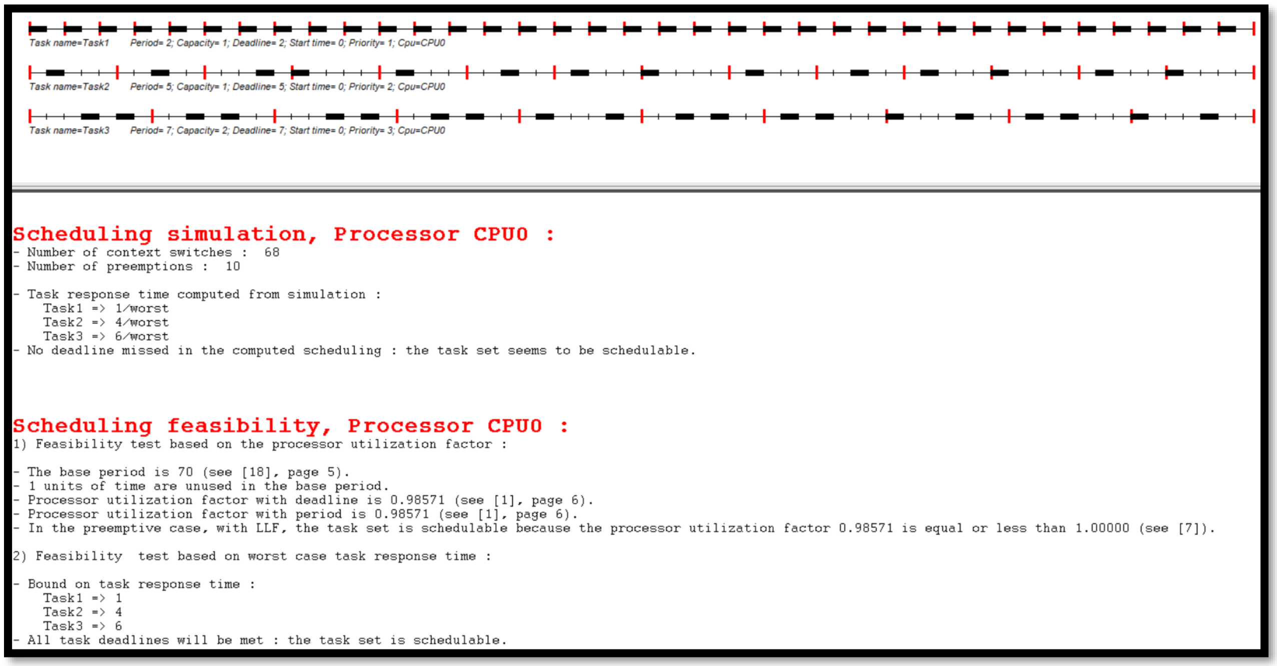
EARLIEST DEADLINE FIRST ANALYSIS:


Task name=Task1 Period= 2; Capacity= 1; Deadline= 2; Start time= 0; Priority= 1; Cpu=CPU0
Task name=Task2 Period= 5; Capacity= 1; Deadline= 5; Start time= 0; Priority= 2; Cpu=CPU0
Task name=Task3 Period= 7; Capacity= 2; Deadline= 7; Start time= 0; Priority= 3; Cpu=CPU0

```
Scheduling simulation, Processor CPU0 :
- Number of context switches :  68
- Number of preemptions :  10

- Task response time computed from simulation :
    Task1 => 1/worst
    Task2 => 4/worst
    Task3 => 6/worst
- No deadline missed in the computed scheduling : the task set seems to be schedulable.



Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The base period is 70 (see [18], page 5).
- 1 units of time are unused in the base period.
- Processor utilization factor with deadline is 0.98571 (see [1], page 6).
- Processor utilization factor with period is 0.98571 (see [1], page 6).
- In the preemptive case, with EDF, the task set is schedulable because the processor utilization factor 0.98571 is equal or less than 1.00000 (see [1], page 8,
theorem 2).


2) Feasibility  test based on worst case task response time :

- Bound on task response time :
    Task1 => 1
    Task2 => 4
    Task3 => 6
- All task deadlines will be met : the task set is schedulable.
```

Figure 11

LEAST LAXITY FIRST ANALYSIS:


Task name=Task1 Period= 2; Capacity= 1; Deadline= 2; Start time= 0; Priority= 1; Cpu=CPU0
Task name=Task2 Period= 5; Capacity= 1; Deadline= 5; Start time= 0; Priority= 2; Cpu=CPU0
Task name=Task3 Period= 7; Capacity= 2; Deadline= 7; Start time= 0; Priority= 3; Cpu=CPU0

```
Scheduling simulation, Processor CPU0 :
- Number of context switches :  68
- Number of preemptions :  10

- Task response time computed from simulation :
    Task1 => 1/worst
    Task2 => 4/worst
    Task3 => 6/worst
- No deadline missed in the computed scheduling : the task set seems to be schedulable.


Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The base period is 70 (see [18], page 5).
- 1 units of time are unused in the base period.
- Processor utilization factor with deadline is 0.98571 (see [1], page 6).
- Processor utilization factor with period is 0.98571 (see [1], page 6).
- In the preemptive case, with LLF, the task set is schedulable because the processor utilization factor 0.98571 is equal or less than 1.00000 (see [7]).

2) Feasibility  test based on worst case task response time :

- Bound on task response time :
    Task1 => 1
    Task2 => 4
    Task3 => 6
- All task deadlines will be met : the task set is schedulable.
```

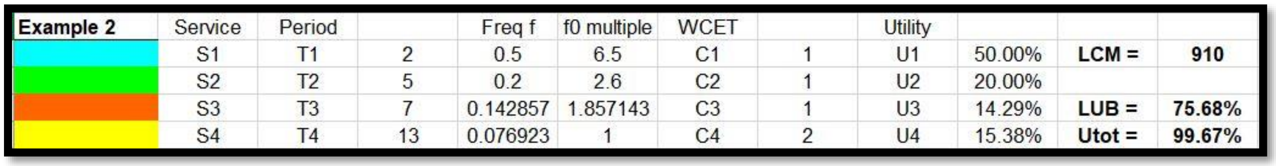
Figure 12

<u>EXPLANATION FOR SERVICE SET 1</u>: From figure 10 and 32, it can be seen that the task set cannot be scheduled according to RM policy as Task 3 tends to miss its deadline over the period of observation (LCM of time periods of all tasks in the task set). The output of the code mentions that the service set is Infeasible, meaning that the task set failed to pass the completion time test and the scheduling point test. This was confirmed by cheddar analysis for that service set. The rate monotonic test depicts that the task set is not schedulable as some tasks tend to miss their deadline. It can be observed that the task set can be scheduled successfully by adapting a different scheduling policy like the earliest deadline first or the least laxity first. This is because in RM policy, the task with higher priority always gets the CPU resource required for its completion. By following this scheduling policy, this task set cannot be scheduled as task 3, being the least priority task, misses its deadline. On the other hand, if Earliest deadline first policy is adapted, then task 3 can also be successfully scheduled and the task set becomes schedulable.

SERVICE SET 2

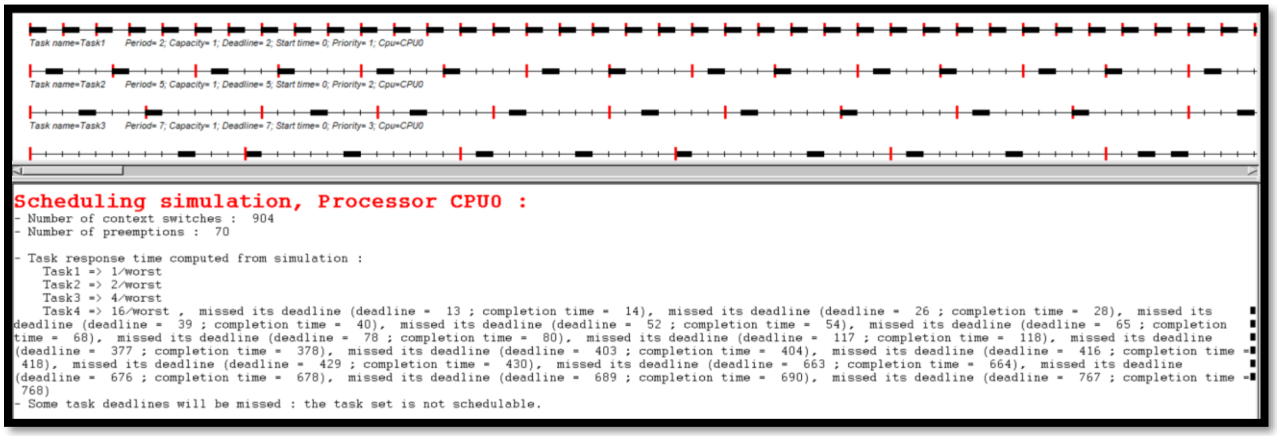
Time Periods: T1=2 T2=5 T3=7 T4=13

Run-Times: C1=1 C2=1 C3=1 C4=2



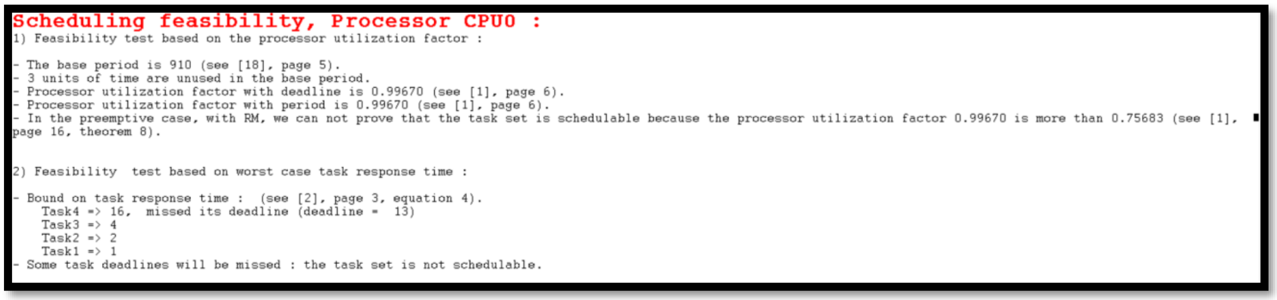
Example 2	Service	Period		Freq f	f0 multiple	WCET		Utility			
🟦	S1	T1	2	0.5	6.5	C1	1	U1	50.00%	**LCM =**	**910**
🟩	S2	T2	5	0.2	2.6	C2	1	U2	20.00%		
🟧	S3	T3	7	0.142857	1.857143	C3	1	U3	14.29%	**LUB =**	**75.68%**
🟨	S4	T4	13	0.076923	1	C4	2	U4	15.38%	**Utot =**	**99.67%**

RATE MONOTONIC ANALYSIS:


Task name=Task1 Period= 2; Capacity= 1; Deadline= 2; Start time= 0; Priority= 1; Cpu=CPU0
Task name=Task2 Period= 5; Capacity= 1; Deadline= 5; Start time= 0; Priority= 2; Cpu=CPU0
Task name=Task3 Period= 7; Capacity= 1; Deadline= 7; Start time= 0; Priority= 3; Cpu=CPU0

```
Scheduling simulation, Processor CPU0 :
- Number of context switches :  904
- Number of preemptions :  70

- Task response time computed from simulation :
    Task1 => 1/worst
    Task2 => 2/worst
    Task3 => 4/worst
    Task4 => 16/worst ,  missed its deadline (deadline =  13 ; completion time =  14),  missed its deadline (deadline =  26 ; completion time =  28),  missed its
deadline (deadline =  39 ; completion time =  40),  missed its deadline (deadline =  52 ; completion time =  54),  missed its deadline (deadline =  65 ; completion
time =  68),  missed its deadline (deadline =  78 ; completion time =  80),  missed its deadline (deadline =  117 ; completion time =  118),  missed its deadline
(deadline =  377 ; completion time =  378),  missed its deadline (deadline =  403 ; completion time =  404),  missed its deadline (deadline =  416 ; completion time =
 418),  missed its deadline (deadline =  429 ; completion time =  430),  missed its deadline (deadline =  663 ; completion time =  664),  missed its deadline
(deadline =  676 ; completion time =  678),  missed its deadline (deadline =  689 ; completion time =  690),  missed its deadline (deadline =  767 ; completion time =
 768)
- Some task deadlines will be missed : the task set is not schedulable.
```

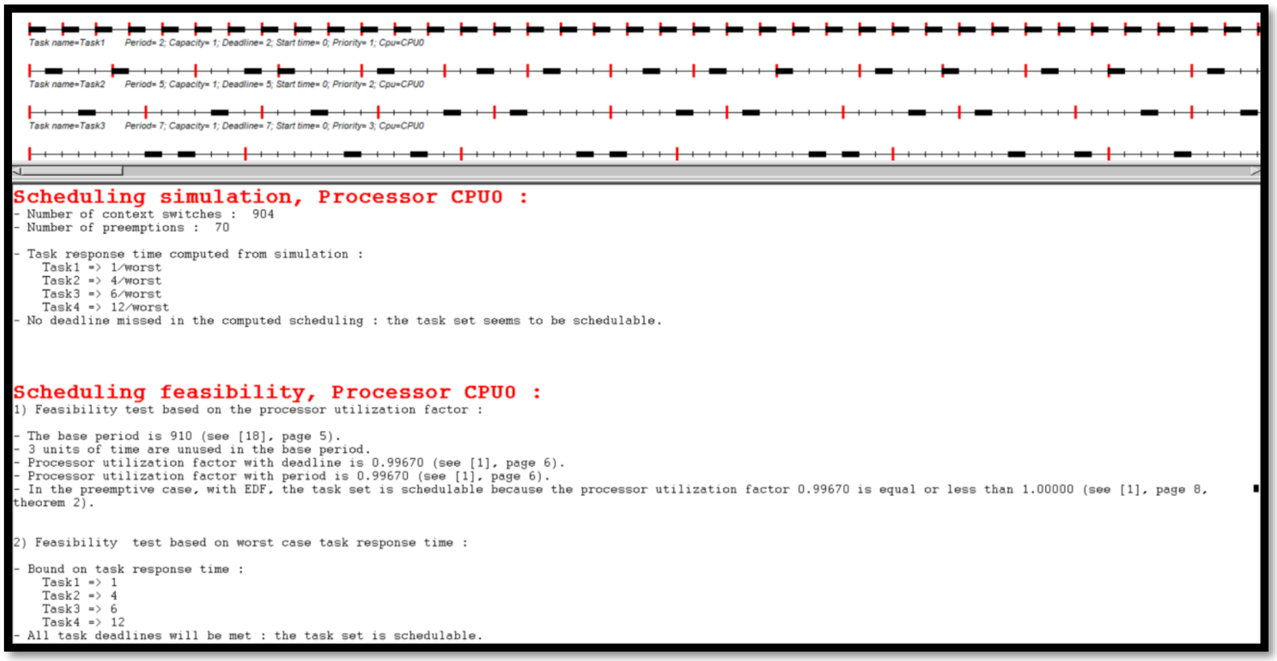
Figure 13



```
Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The base period is 910 (see [18], page 5).
- 3 units of time are unused in the base period.
- Processor utilization factor with deadline is 0.99670 (see [1], page 6).
- Processor utilization factor with period is 0.99670 (see [1], page 6).
- In the preemptive case, with RM, we can not prove that the task set is schedulable because the processor utilization factor 0.99670 is more than 0.75683 (see [1],
page 16, theorem 8).


2) Feasibility  test based on worst case task response time :

- Bound on task response time :  (see [2], page 3, equation 4).
    Task4 => 16,  missed its deadline (deadline =  13)
    Task3 => 4
    Task2 => 2
    Task1 => 1
- Some task deadlines will be missed : the task set is not schedulable.
```

Figure 14

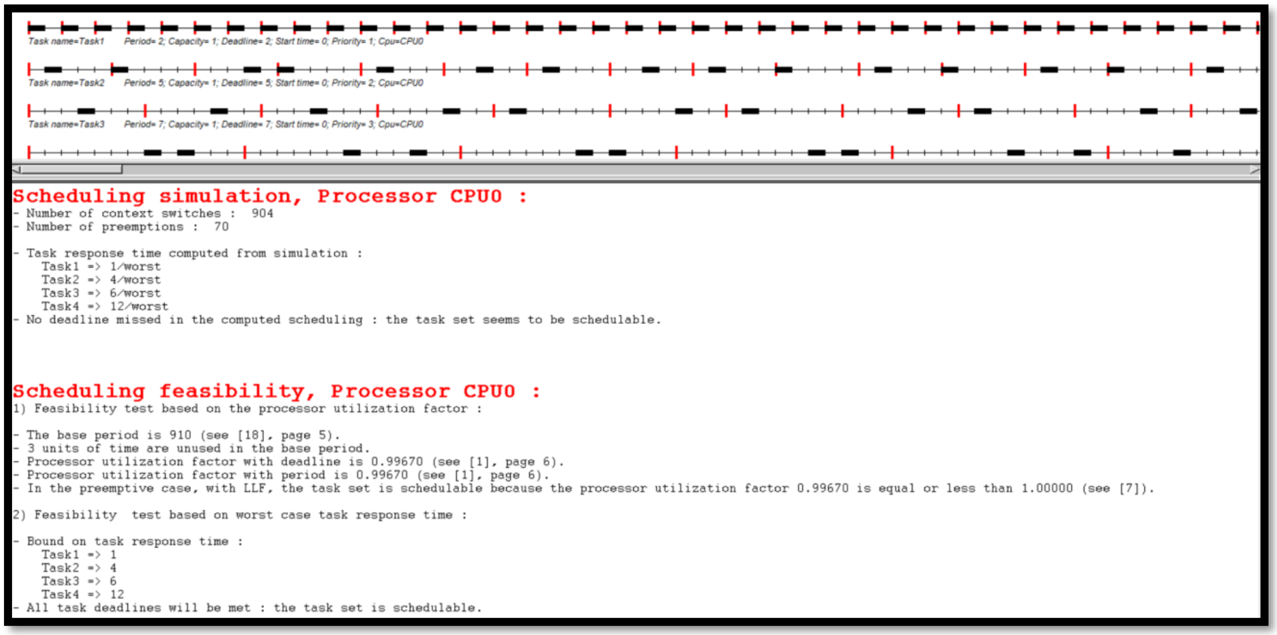
<u>EARLIEST DEADLINE FIRST ANALYSIS:</u>


Task name=Task1 Period= 2; Capacity= 1; Deadline= 2; Start time= 0; Priority= 1; Cpu=CPU0
Task name=Task2 Period= 5; Capacity= 1; Deadline= 5; Start time= 0; Priority= 2; Cpu=CPU0
Task name=Task3 Period= 7; Capacity= 1; Deadline= 7; Start time= 0; Priority= 3; Cpu=CPU0

```
Scheduling simulation, Processor CPU0 :
- Number of context switches :  904
- Number of preemptions :  70

- Task response time computed from simulation :
    Task1 => 1/worst
    Task2 => 4/worst
    Task3 => 6/worst
    Task4 => 12/worst
- No deadline missed in the computed scheduling : the task set seems to be schedulable.


Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The base period is 910 (see [18], page 5).
- 3 units of time are unused in the base period.
- Processor utilization factor with deadline is 0.99670 (see [1], page 6).
- Processor utilization factor with period is 0.99670 (see [1], page 6).
- In the preemptive case, with EDF, the task set is schedulable because the processor utilization factor 0.99670 is equal or less than 1.00000 (see [1], page 8,
theorem 2).


2) Feasibility  test based on worst case task response time :

- Bound on task response time :
    Task1 => 1
    Task2 => 4
    Task3 => 6
    Task4 => 12
- All task deadlines will be met : the task set is schedulable.
```

Figure 15

<u>LEAST LAXITY FIRST ANALYSIS:</u>


Task name=Task1 Period= 2; Capacity= 1; Deadline= 2; Start time= 0; Priority= 1; Cpu=CPU0
Task name=Task2 Period= 5; Capacity= 1; Deadline= 5; Start time= 0; Priority= 2; Cpu=CPU0
Task name=Task3 Period= 7; Capacity= 1; Deadline= 7; Start time= 0; Priority= 3; Cpu=CPU0

```
Scheduling simulation, Processor CPU0 :
- Number of context switches :  904
- Number of preemptions :  70

- Task response time computed from simulation :
    Task1 => 1/worst
    Task2 => 4/worst
    Task3 => 6/worst
    Task4 => 12/worst
- No deadline missed in the computed scheduling : the task set seems to be schedulable.


Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The base period is 910 (see [18], page 5).
- 3 units of time are unused in the base period.
- Processor utilization factor with deadline is 0.99670 (see [1], page 6).
- Processor utilization factor with period is 0.99670 (see [1], page 6).
- In the preemptive case, with LLF, the task set is schedulable because the processor utilization factor 0.99670 is equal or less than 1.00000 (see [7]).

2) Feasibility  test based on worst case task response time :

- Bound on task response time :
    Task1 => 1
    Task2 => 4
    Task3 => 6
    Task4 => 12
- All task deadlines will be met : the task set is schedulable.
```

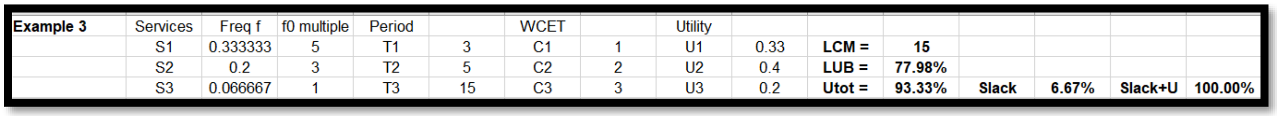
Figure 16

<u>EXPLANATION FOR SERVICE SET 2</u>: It can be observed that this task set fails to satisfy the completion time test and the scheduling point test. Thus, this task is not feasible. However, if this task set is scheduled using EDF or LLF policy, then this task set can be scheduled.

SERVICE SET 3

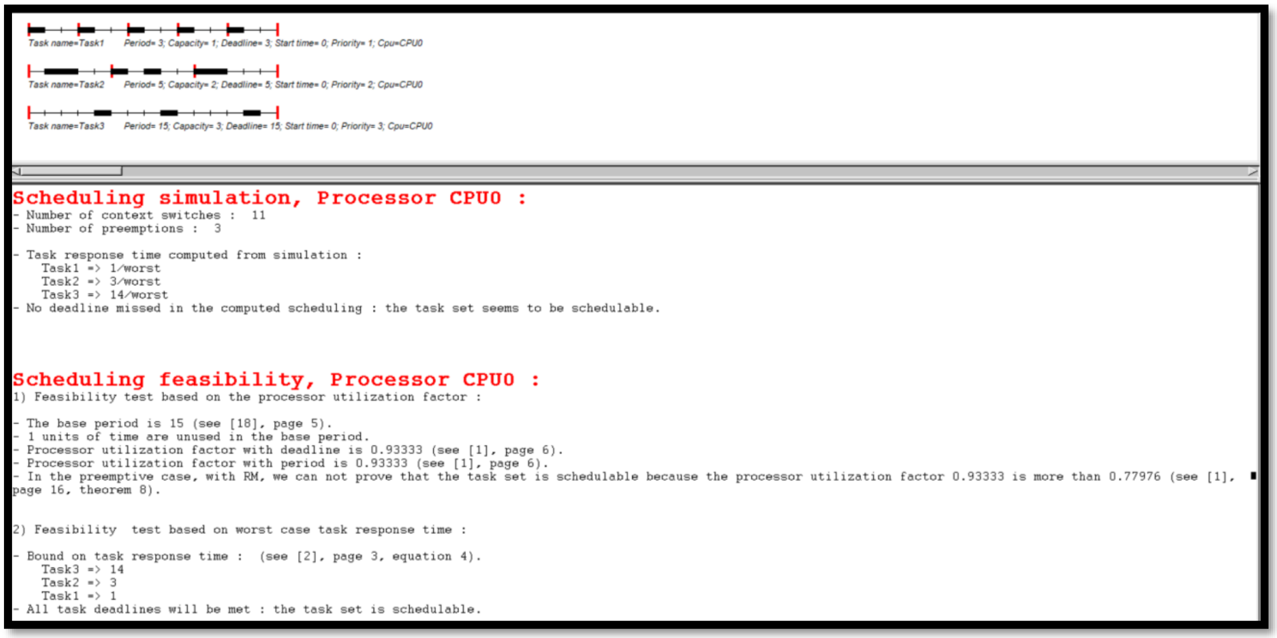
Time Periods: T1=3 T2=5 T3=15

Run-Times: C1=1 C2=2 C3=3



Example 3	Services	Freq f	f0 multiple	Period		WCET		Utility							
	S1	0.333333	5	T1	3	C1	1	U1	0.33	LCM =	15				
	S2	0.2	3	T2	5	C2	2	U2	0.4	LUB =	77.98%				
	S3	0.066667	1	T3	15	C3	3	U3	0.2	Utot =	93.33%	Slack	6.67%	Slack+U	100.00%

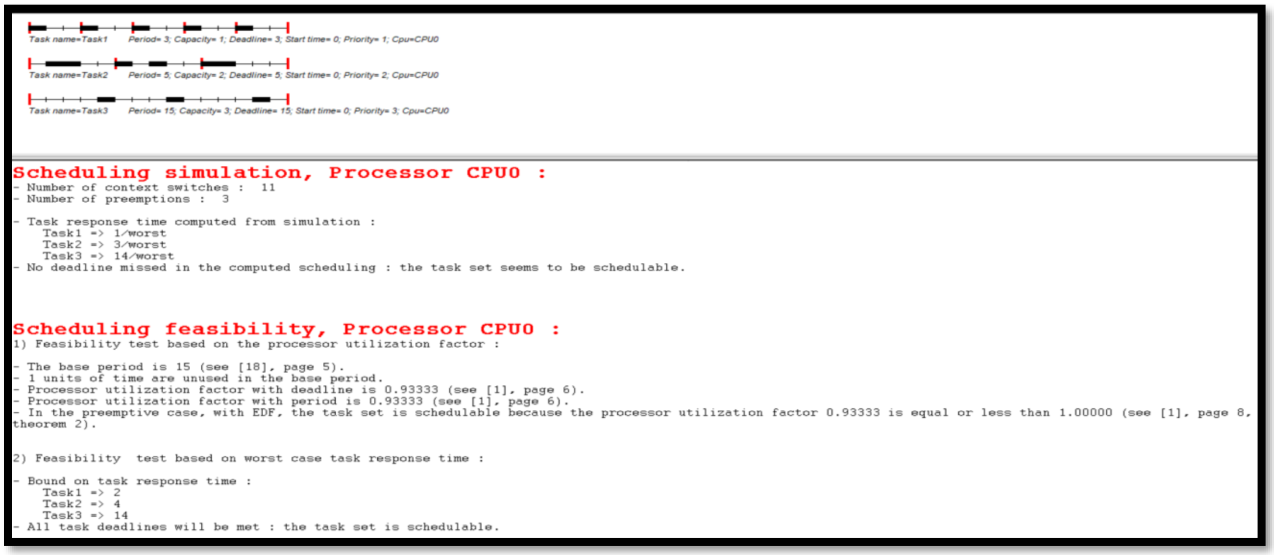
RATE MONOTONIC ANALYSIS:



```
Task name=Task1    Period= 3; Capacity= 1; Deadline= 3; Start time= 0; Priority= 1; Cpu=CPU0
Task name=Task2    Period= 5; Capacity= 2; Deadline= 5; Start time= 0; Priority= 2; Cpu=CPU0
Task name=Task3    Period= 15; Capacity= 3; Deadline= 15; Start time= 0; Priority= 3; Cpu=CPU0

Scheduling simulation, Processor CPU0 :
- Number of context switches :  11
- Number of preemptions :  3

- Task response time computed from simulation :
    Task1 => 1/worst
    Task2 => 3/worst
    Task3 => 14/worst
- No deadline missed in the computed scheduling : the task set seems to be schedulable.


Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The base period is 15 (see [18], page 5).
- 1 units of time are unused in the base period.
- Processor utilization factor with deadline is 0.93333 (see [1], page 6).
- Processor utilization factor with period is 0.93333 (see [1], page 6).
- In the preemptive case, with RM, we can not prove that the task set is schedulable because the processor utilization factor 0.93333 is more than 0.77976 (see [1],
page 16, theorem 8).


2) Feasibility  test based on worst case task response time :

- Bound on task response time :  (see [2], page 3, equation 4).
    Task3 => 14
    Task2 => 3
    Task1 => 1
- All task deadlines will be met : the task set is schedulable.
```

Figure 17

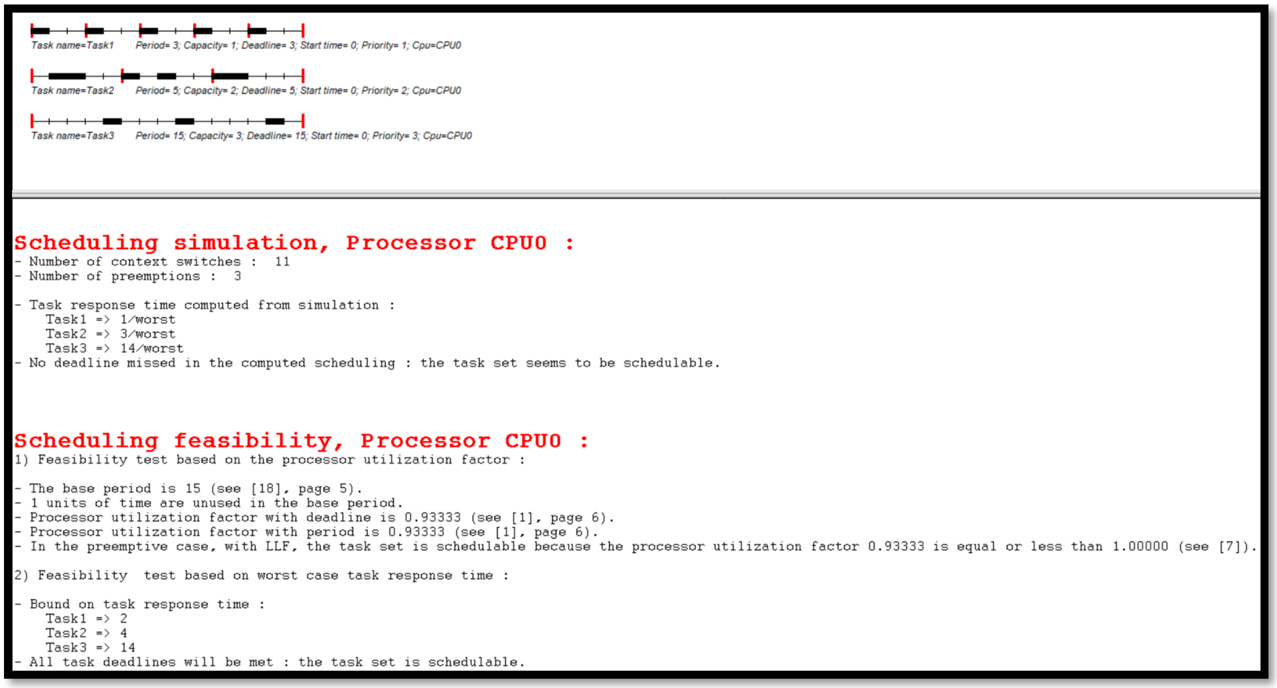
EARLIEST DEADLINE FIRST ANALYSIS:



```
Task name=Task1    Period= 3; Capacity= 1; Deadline= 3; Start time= 0; Priority= 1; Cpu=CPU0
Task name=Task2    Period= 5; Capacity= 2; Deadline= 5; Start time= 0; Priority= 2; Cpu=CPU0
Task name=Task3    Period= 15; Capacity= 3; Deadline= 15; Start time= 0; Priority= 3; Cpu=CPU0

Scheduling simulation, Processor CPU0 :
- Number of context switches :  11
- Number of preemptions :  3

- Task response time computed from simulation :
    Task1 => 1/worst
    Task2 => 3/worst
    Task3 => 14/worst
- No deadline missed in the computed scheduling : the task set seems to be schedulable.


Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The base period is 15 (see [18], page 5).
- 1 units of time are unused in the base period.
- Processor utilization factor with deadline is 0.93333 (see [1], page 6).
- Processor utilization factor with period is 0.93333 (see [1], page 6).
- In the preemptive case, with EDF, the task set is schedulable because the processor utilization factor 0.93333 is equal or less than 1.00000 (see [1], page 8,
theorem 2).

2) Feasibility  test based on worst case task response time :

- Bound on task response time :
    Task1 => 2
    Task2 => 4
    Task3 => 14
- All task deadlines will be met : the task set is schedulable.
```

Figure 18

LEAST LAXITY FIRST ANALYSIS:


Task name=Task1 Period= 3; Capacity= 1; Deadline= 3; Start time= 0; Priority= 1; Cpu=CPU0
Task name=Task2 Period= 5; Capacity= 2; Deadline= 5; Start time= 0; Priority= 2; Cpu=CPU0
Task name=Task3 Period= 15; Capacity= 3; Deadline= 15; Start time= 0; Priority= 3; Cpu=CPU0

```
Scheduling simulation, Processor CPU0 :
- Number of context switches :  11
- Number of preemptions :  3

- Task response time computed from simulation :
    Task1 => 1/worst
    Task2 => 3/worst
    Task3 => 14/worst
- No deadline missed in the computed scheduling : the task set seems to be schedulable.


Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The base period is 15 (see [18], page 5).
- 1 units of time are unused in the base period.
- Processor utilization factor with deadline is 0.93333 (see [1], page 6).
- Processor utilization factor with period is 0.93333 (see [1], page 6).
- In the preemptive case, with LLF, the task set is schedulable because the processor utilization factor 0.93333 is equal or less than 1.00000 (see [7]).

2) Feasibility  test based on worst case task response time :

- Bound on task response time :
    Task1 => 2
    Task2 => 4
    Task3 => 14
- All task deadlines will be met : the task set is schedulable.
```

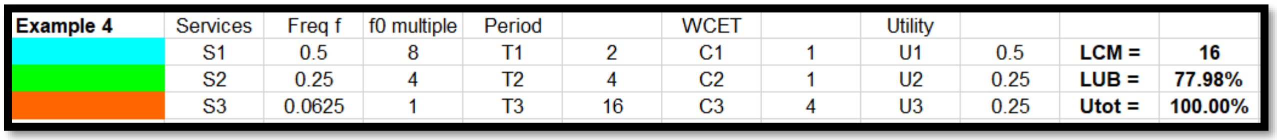
Figure 19

<u>EXPLANATION FOR SERVICE SET 3</u>: It can be observed that this task set is successful in passing the completion time test and the scheduling point test. Thus, this task is feasible. This task set can also be scheduled using EDF or LLF policy.

SERVICE SET 4

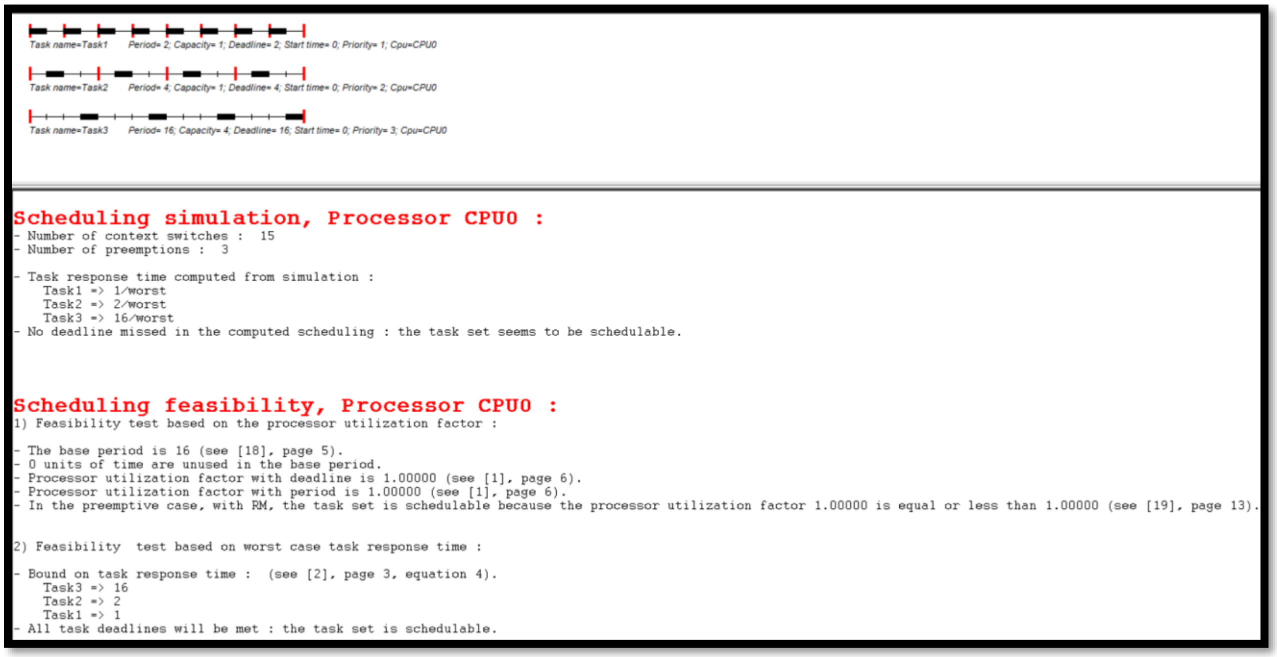
Time Periods: T1=2 T2=4 T3=16

Run-Times: C1=1 C2=1 C3=4



Example 4	Services	Freq f	f0 multiple	Period		WCET		Utility			
🟦	S1	0.5	8	T1	2	C1	1	U1	0.5	**LCM =**	**16**
🟩	S2	0.25	4	T2	4	C2	1	U2	0.25	**LUB =**	**77.98%**
🟧	S3	0.0625	1	T3	16	C3	4	U3	0.25	**Utot =**	**100.00%**

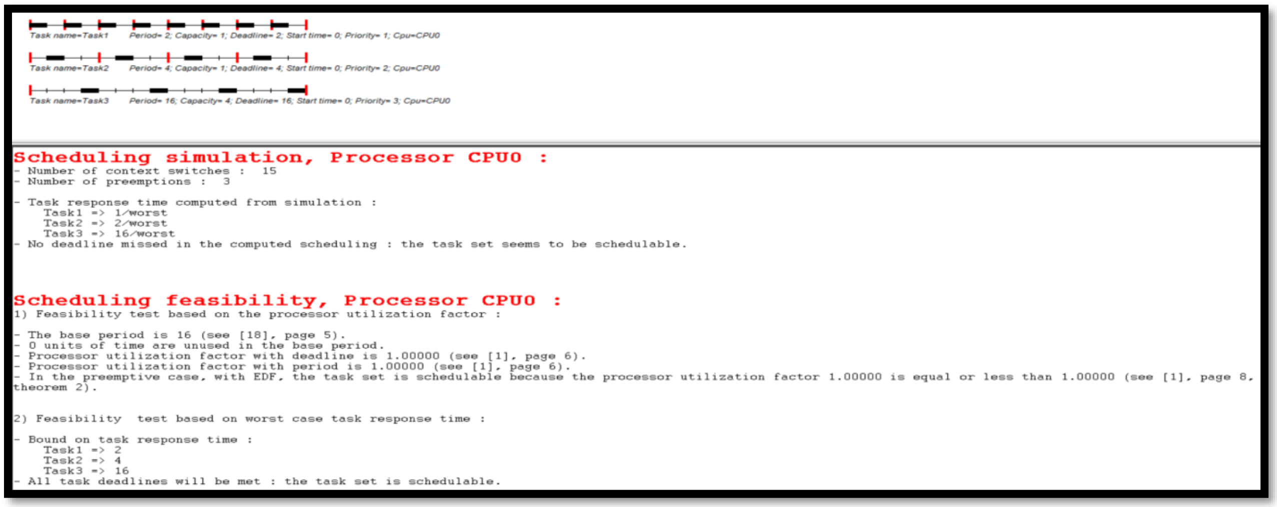
RATE MONOTONIC ANALYSIS:



```
Task name=Task1     Period= 2; Capacity= 1; Deadline= 2; Start time= 0; Priority= 1; Cpu=CPU0
Task name=Task2     Period= 4; Capacity= 1; Deadline= 4; Start time= 0; Priority= 2; Cpu=CPU0
Task name=Task3     Period= 16; Capacity= 4; Deadline= 16; Start time= 0; Priority= 3; Cpu=CPU0

Scheduling simulation, Processor CPU0 :
- Number of context switches :  15
- Number of preemptions :  3

- Task response time computed from simulation :
    Task1 => 1/worst
    Task2 => 2/worst
    Task3 => 16/worst
- No deadline missed in the computed scheduling : the task set seems to be schedulable.


Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The base period is 16 (see [18], page 5).
- 0 units of time are unused in the base period.
- Processor utilization factor with deadline is 1.00000 (see [1], page 6).
- Processor utilization factor with period is 1.00000 (see [1], page 6).
- In the preemptive case, with RM, the task set is schedulable because the processor utilization factor 1.00000 is equal or less than 1.00000 (see [19], page 13).

2) Feasibility  test based on worst case task response time :

- Bound on task response time :  (see [2], page 3, equation 4).
    Task3 => 16
    Task2 => 2
    Task1 => 1
- All task deadlines will be met : the task set is schedulable.
```

Figure 20

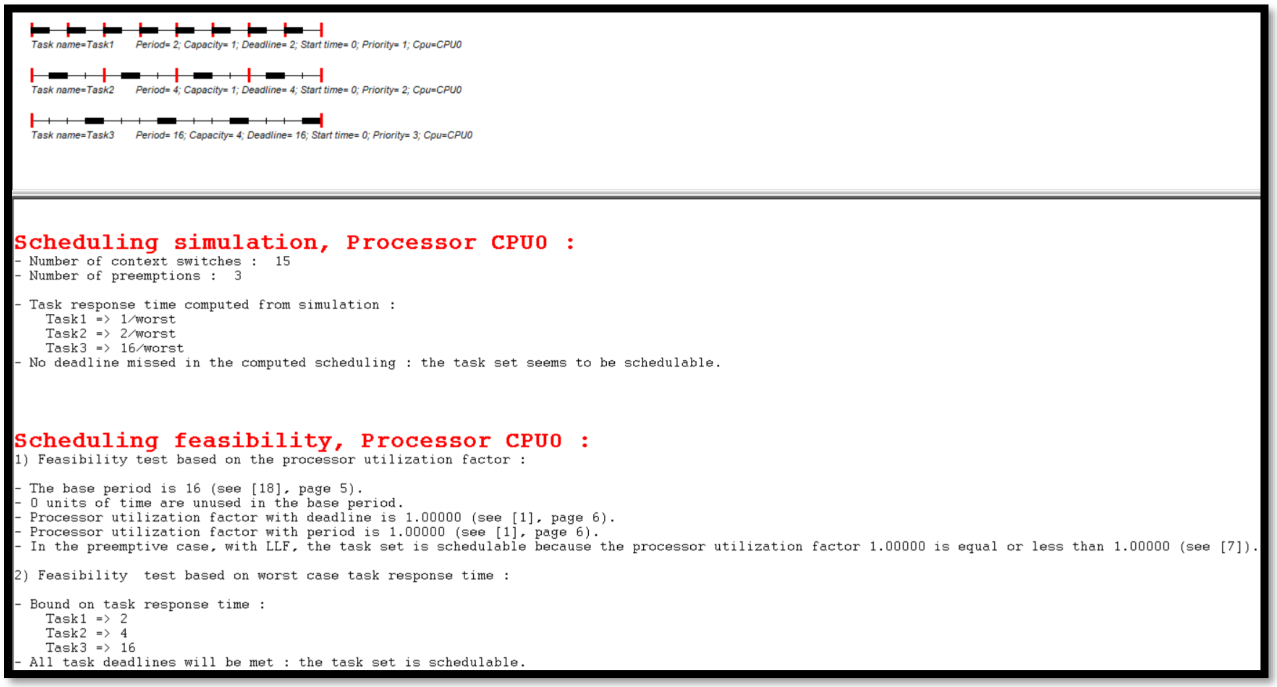
EARLIEST DEADLINE FIRST ANALYSIS:



```
Task name=Task1     Period= 2; Capacity= 1; Deadline= 2; Start time= 0; Priority= 1; Cpu=CPU0
Task name=Task2     Period= 4; Capacity= 1; Deadline= 4; Start time= 0; Priority= 2; Cpu=CPU0
Task name=Task3     Period= 16; Capacity= 4; Deadline= 16; Start time= 0; Priority= 3; Cpu=CPU0

Scheduling simulation, Processor CPU0 :
- Number of context switches :  15
- Number of preemptions :  3

- Task response time computed from simulation :
    Task1 => 1/worst
    Task2 => 2/worst
    Task3 => 16/worst
- No deadline missed in the computed scheduling : the task set seems to be schedulable.


Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The base period is 16 (see [18], page 5).
- 0 units of time are unused in the base period.
- Processor utilization factor with deadline is 1.00000 (see [1], page 6).
- Processor utilization factor with period is 1.00000 (see [1], page 6).
- In the preemptive case, with EDF, the task set is schedulable because the processor utilization factor 1.00000 is equal or less than 1.00000 (see [1], page 8,
theorem 2).

2) Feasibility  test based on worst case task response time :

- Bound on task response time :
    Task1 => 2
    Task2 => 4
    Task3 => 16
- All task deadlines will be met : the task set is schedulable.
```

Figure 21

LEAST LAXITY FIRST ANALYSIS:


Task name=Task1 Period= 2; Capacity= 1; Deadline= 2; Start time= 0; Priority= 1; Cpu=CPU0
Task name=Task2 Period= 4; Capacity= 1; Deadline= 4; Start time= 0; Priority= 2; Cpu=CPU0
Task name=Task3 Period= 16; Capacity= 4; Deadline= 16; Start time= 0; Priority= 3; Cpu=CPU0

```
Scheduling simulation, Processor CPU0 :
- Number of context switches :  15
- Number of preemptions :  3

- Task response time computed from simulation :
    Task1 => 1/worst
    Task2 => 2/worst
    Task3 => 16/worst
- No deadline missed in the computed scheduling : the task set seems to be schedulable.


Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The base period is 16 (see [18], page 5).
- 0 units of time are unused in the base period.
- Processor utilization factor with deadline is 1.00000 (see [1], page 6).
- Processor utilization factor with period is 1.00000 (see [1], page 6).
- In the preemptive case, with LLF, the task set is schedulable because the processor utilization factor 1.00000 is equal or less than 1.00000 (see [7]).

2) Feasibility  test based on worst case task response time :

- Bound on task response time :
    Task1 => 2
    Task2 => 4
    Task3 => 16
- All task deadlines will be met : the task set is schedulable.
```

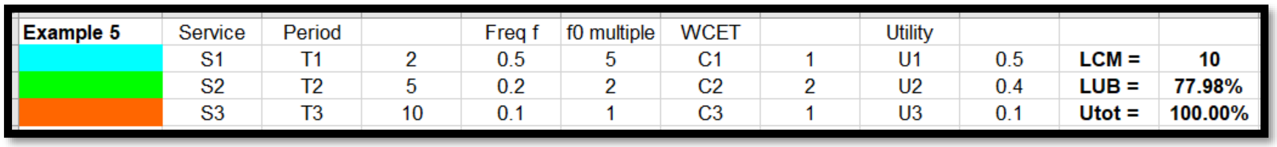
Figure 22

<u>EXPLANATION FOR SERVICE SET 4</u>: It can be observed that this task set is successful in passing the completion time test and the scheduling point test. Thus, this task is feasible. This task set can also be scheduled using EDF or LLF policy.

SERVICE SET 5

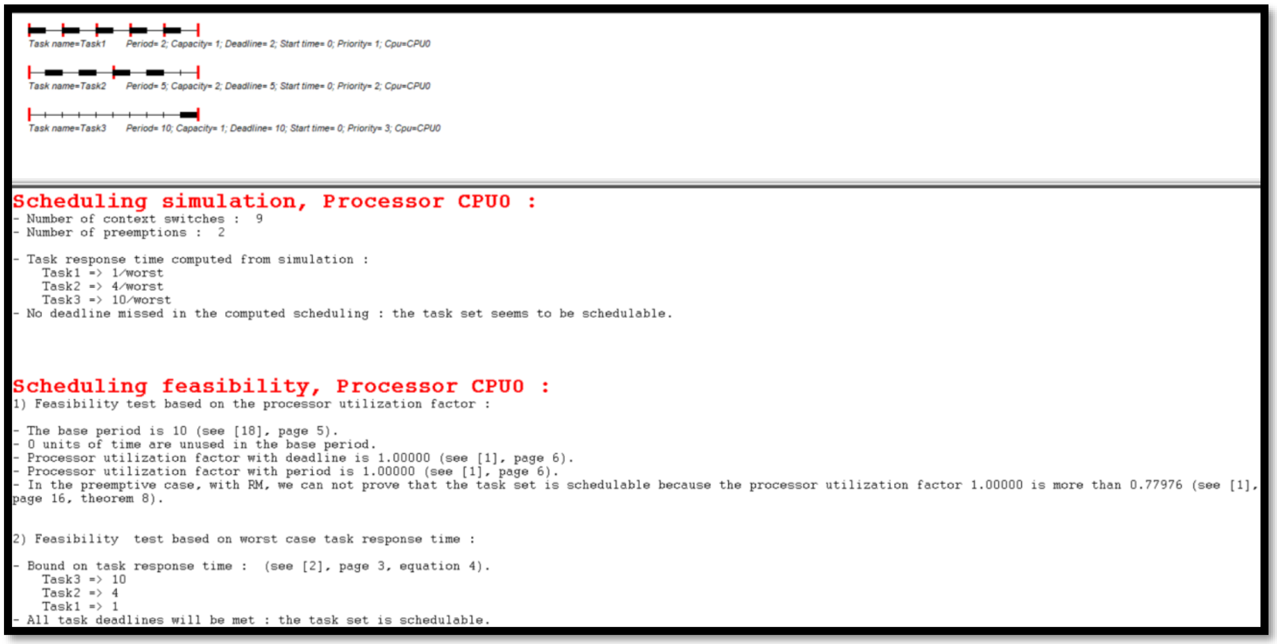
Time Periods: T1=2 T2=5 T3=10

Run-Times: C1=1 C2=2 C3=1



Example 5	Service	Period		Freq f	f0 multiple	WCET		Utility			
🟦	S1	T1	2	0.5	5	C1	1	U1	0.5	**LCM =**	**10**
🟩	S2	T2	5	0.2	2	C2	2	U2	0.4	**LUB =**	**77.98%**
🟧	S3	T3	10	0.1	1	C3	1	U3	0.1	**Utot =**	**100.00%**

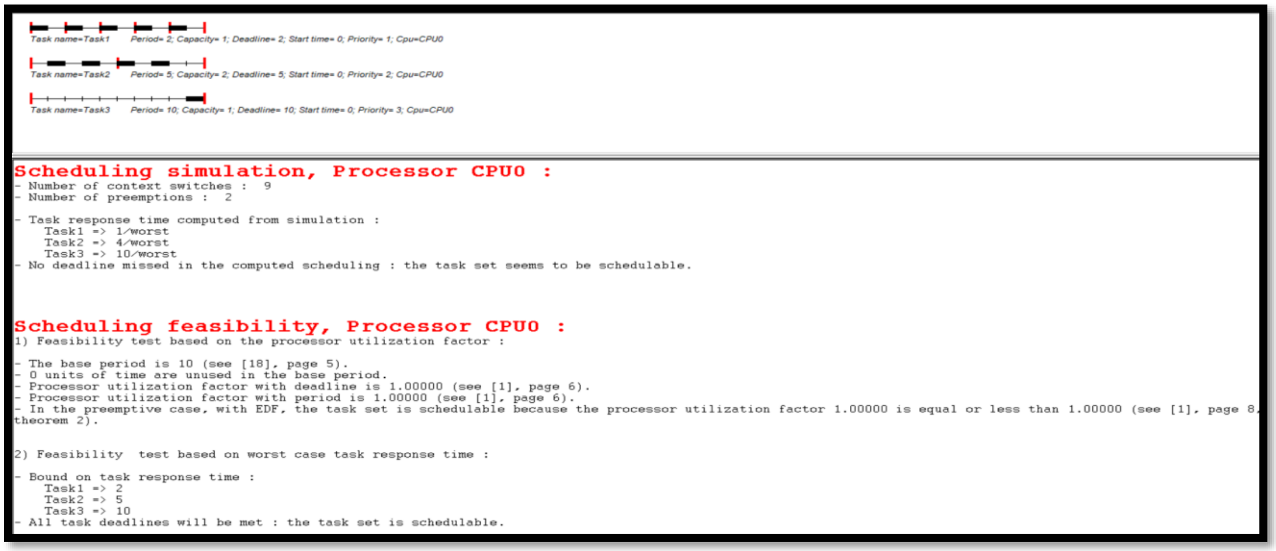
RATE MONOTONIC ANALYSIS:



```
Task name=Task1     Period= 2; Capacity= 1; Deadline= 2; Start time= 0; Priority= 1; Cpu=CPU0
Task name=Task2     Period= 5; Capacity= 2; Deadline= 5; Start time= 0; Priority= 2; Cpu=CPU0
Task name=Task3     Period= 10; Capacity= 1; Deadline= 10; Start time= 0; Priority= 3; Cpu=CPU0

Scheduling simulation, Processor CPU0 :
- Number of context switches :  9
- Number of preemptions :  2

- Task response time computed from simulation :
    Task1 => 1/worst
    Task2 => 4/worst
    Task3 => 10/worst
- No deadline missed in the computed scheduling : the task set seems to be schedulable.


Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The base period is 10 (see [18], page 5).
- 0 units of time are unused in the base period.
- Processor utilization factor with deadline is 1.00000 (see [1], page 6).
- Processor utilization factor with period is 1.00000 (see [1], page 6).
- In the preemptive case, with RM, we can not prove that the task set is schedulable because the processor utilization factor 1.00000 is more than 0.77976 (see [1],
page 16, theorem 8).


2) Feasibility  test based on worst case task response time :

- Bound on task response time :  (see [2], page 3, equation 4).
    Task3 => 10
    Task2 => 4
    Task1 => 1
- All task deadlines will be met : the task set is schedulable.
```

Figure 23

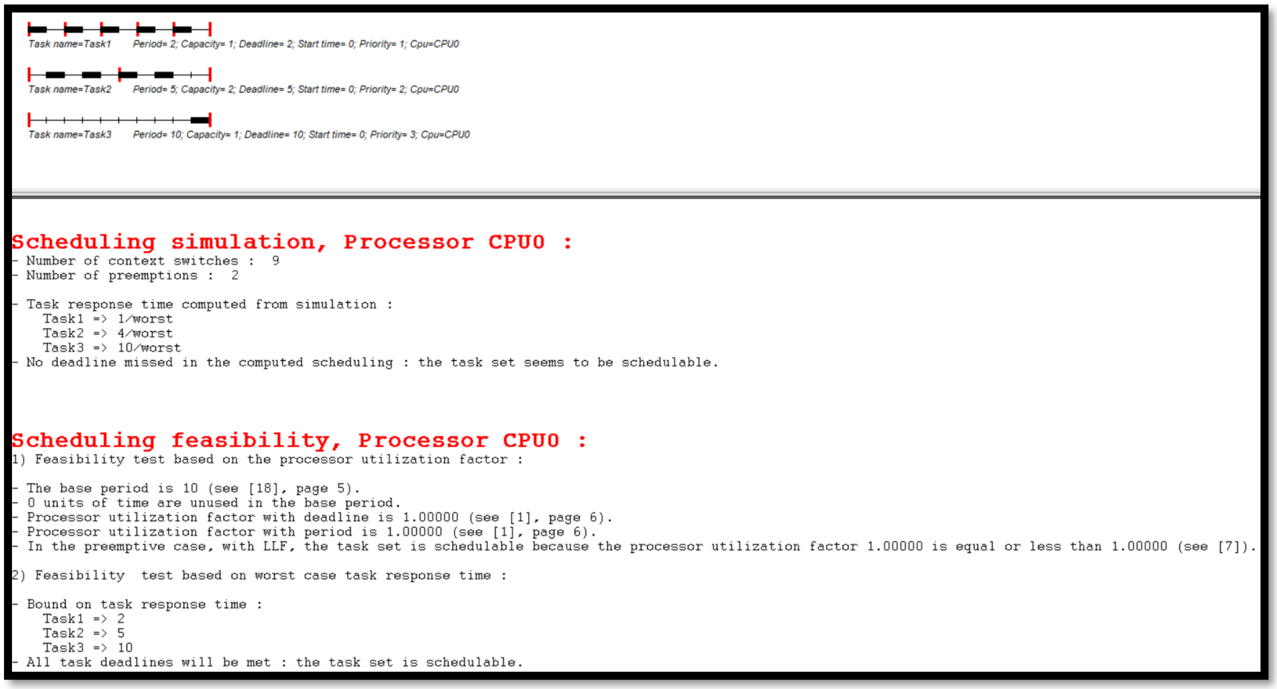
EARLIEST DEADLINE FIRST ANALYSIS:



```
Task name=Task1     Period= 2; Capacity= 1; Deadline= 2; Start time= 0; Priority= 1; Cpu=CPU0
Task name=Task2     Period= 5; Capacity= 2; Deadline= 5; Start time= 0; Priority= 2; Cpu=CPU0
Task name=Task3     Period= 10; Capacity= 1; Deadline= 10; Start time= 0; Priority= 3; Cpu=CPU0

Scheduling simulation, Processor CPU0 :
- Number of context switches :  9
- Number of preemptions :  2

- Task response time computed from simulation :
    Task1 => 1/worst
    Task2 => 4/worst
    Task3 => 10/worst
- No deadline missed in the computed scheduling : the task set seems to be schedulable.


Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The base period is 10 (see [18], page 5).
- 0 units of time are unused in the base period.
- Processor utilization factor with deadline is 1.00000 (see [1], page 6).
- Processor utilization factor with period is 1.00000 (see [1], page 6).
- In the preemptive case, with EDF, the task set is schedulable because the processor utilization factor 1.00000 is equal or less than 1.00000 (see [1], page 8,
theorem 2).

2) Feasibility  test based on worst case task response time :

- Bound on task response time :
    Task1 => 2
    Task2 => 5
    Task3 => 10
- All task deadlines will be met : the task set is schedulable.
```

Figure 24

LEAST LAXITY FIRST ANALYSIS:


Task name=Task1 Period= 2; Capacity= 1; Deadline= 2; Start time= 0; Priority= 1; Cpu=CPU0
Task name=Task2 Period= 5; Capacity= 2; Deadline= 5; Start time= 0; Priority= 2; Cpu=CPU0
Task name=Task3 Period= 10; Capacity= 1; Deadline= 10; Start time= 0; Priority= 3; Cpu=CPU0

```
Scheduling simulation, Processor CPU0 :
- Number of context switches :  9
- Number of preemptions :  2

- Task response time computed from simulation :
    Task1 => 1/worst
    Task2 => 4/worst
    Task3 => 10/worst
- No deadline missed in the computed scheduling : the task set seems to be schedulable.


Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The base period is 10 (see [18], page 5).
- 0 units of time are unused in the base period.
- Processor utilization factor with deadline is 1.00000 (see [1], page 6).
- Processor utilization factor with period is 1.00000 (see [1], page 6).
- In the preemptive case, with LLF, the task set is schedulable because the processor utilization factor 1.00000 is equal or less than 1.00000 (see [7]).

2) Feasibility  test based on worst case task response time :

- Bound on task response time :
    Task1 => 2
    Task2 => 5
    Task3 => 10
- All task deadlines will be met : the task set is schedulable.
```

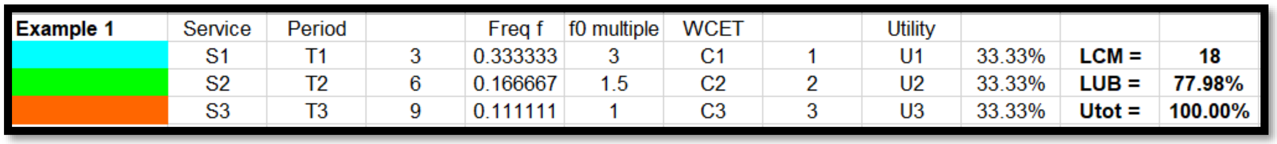
Figure 25

<u>EXPLANATION FOR SERVICE SET 5</u>: It can be observed that this task set is successful in passing the completion time test and the scheduling point test. Thus, this task is feasible. This task set can also be scheduled using EDF or LLF policy.

SERVICE SET 6

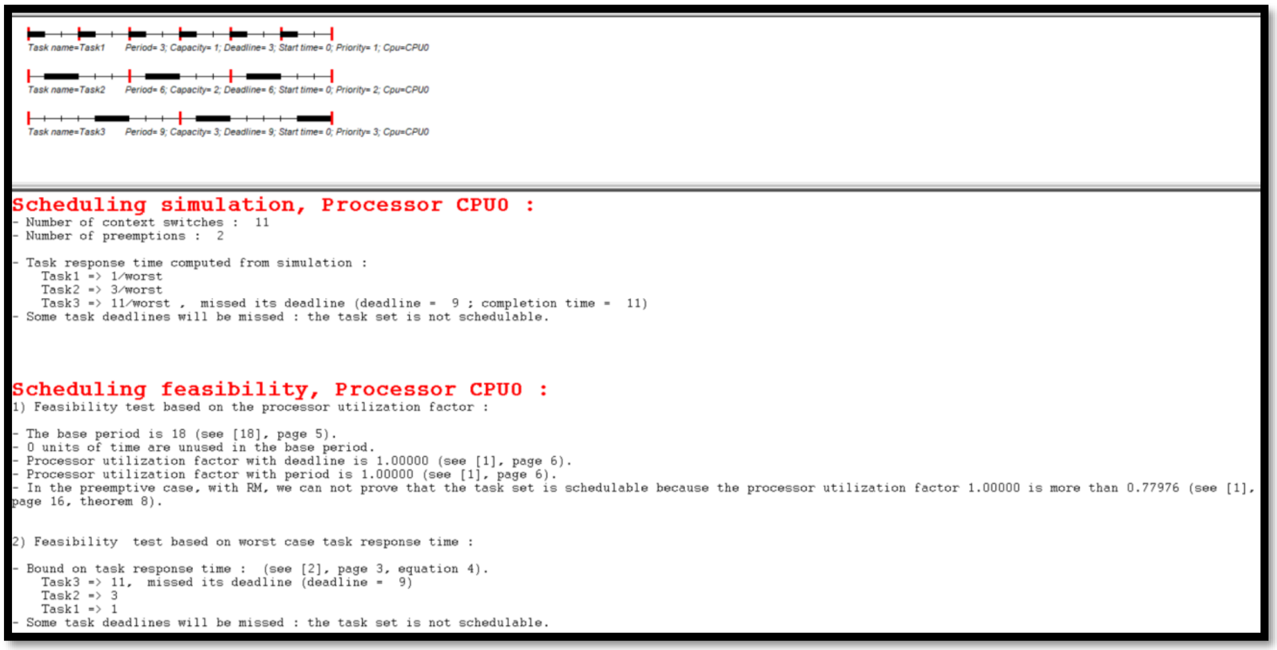
Time Periods: T1=3 T2=6 T3=9

Run-Times: C1=1 C2=2 C3=3



Example 1	Service	Period		Freq f	f0 multiple	WCET		Utility			
🟦	S1	T1	3	0.333333	3	C1	1	U1	33.33%	**LCM =**	**18**
🟩	S2	T2	6	0.166667	1.5	C2	2	U2	33.33%	**LUB =**	**77.98%**
🟧	S3	T3	9	0.111111	1	C3	3	U3	33.33%	**Utot =**	**100.00%**

RATE MONOTONIC ANALYSIS:

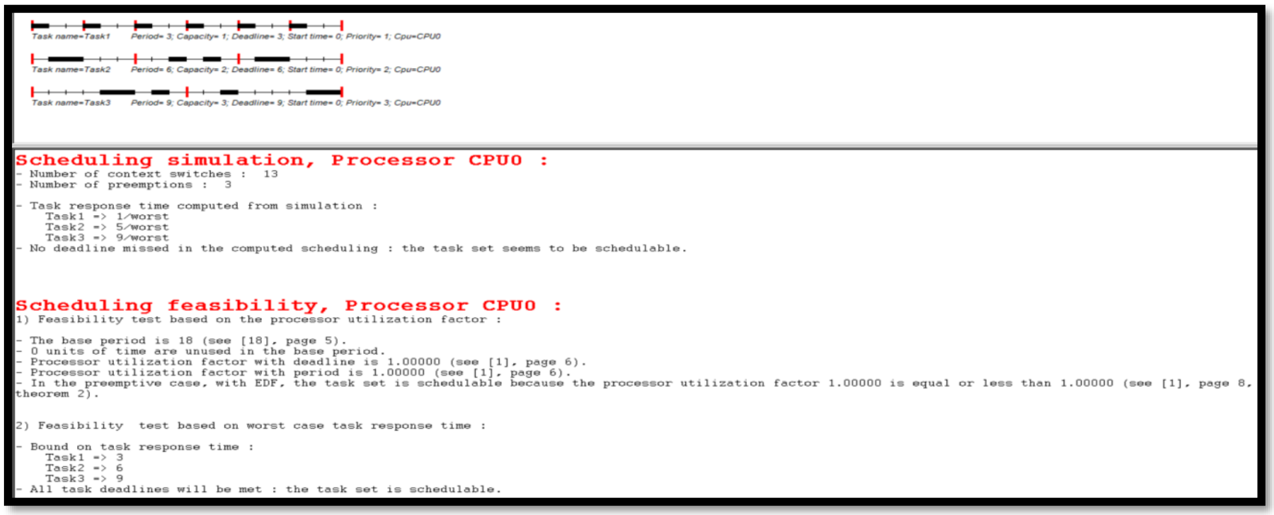


```
Task name=Task1     Period= 3; Capacity= 1; Deadline= 3; Start time= 0; Priority= 1; Cpu=CPU0
Task name=Task2     Period= 6; Capacity= 2; Deadline= 6; Start time= 0; Priority= 2; Cpu=CPU0
Task name=Task3     Period= 9; Capacity= 3; Deadline= 9; Start time= 0; Priority= 3; Cpu=CPU0
```

```
Scheduling simulation, Processor CPU0 :
- Number of context switches :  11
- Number of preemptions :  2

- Task response time computed from simulation :
    Task1 => 1/worst
    Task2 => 3/worst
    Task3 => 11/worst ,  missed its deadline (deadline =  9 ; completion time =  11)
- Some task deadlines will be missed : the task set is not schedulable.


Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The base period is 18 (see [18], page 5).
- 0 units of time are unused in the base period.
- Processor utilization factor with deadline is 1.00000 (see [1], page 6).
- Processor utilization factor with period is 1.00000 (see [1], page 6).
- In the preemptive case, with RM, we can not prove that the task set is schedulable because the processor utilization factor 1.00000 is more than 0.77976 (see [1],
page 16, theorem 8).


2) Feasibility  test based on worst case task response time :

- Bound on task response time :  (see [2], page 3, equation 4).
    Task3 => 11,  missed its deadline (deadline =  9)
    Task2 => 3
    Task1 => 1
- Some task deadlines will be missed : the task set is not schedulable.
```

Figure 26

EARLIEST DEADLINE FIRST ANALYSIS:

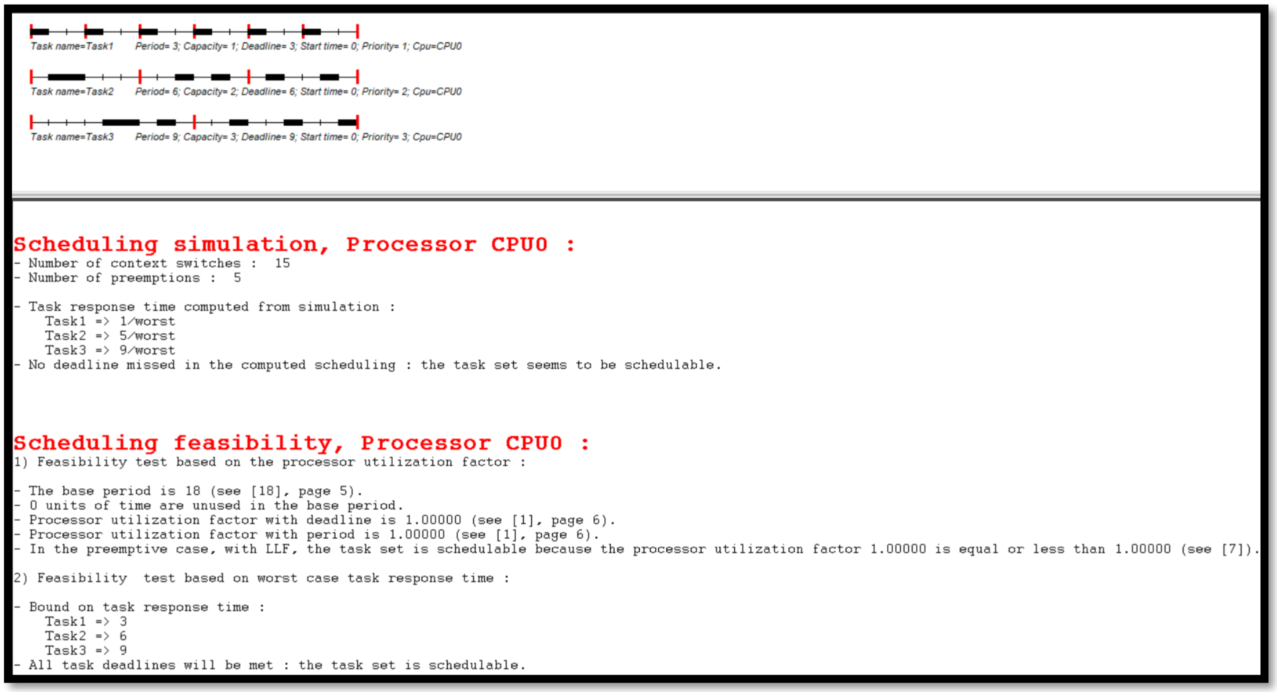


```
Task name=Task1     Period= 3; Capacity= 1; Deadline= 3; Start time= 0; Priority= 1; Cpu=CPU0
Task name=Task2     Period= 6; Capacity= 2; Deadline= 6; Start time= 0; Priority= 2; Cpu=CPU0
Task name=Task3     Period= 9; Capacity= 3; Deadline= 9; Start time= 0; Priority= 3; Cpu=CPU0
```

```
Scheduling simulation, Processor CPU0 :
- Number of context switches :  13
- Number of preemptions :  3

- Task response time computed from simulation :
    Task1 => 1/worst
    Task2 => 5/worst
    Task3 => 9/worst
- No deadline missed in the computed scheduling : the task set seems to be schedulable.


Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The base period is 18 (see [18], page 5).
- 0 units of time are unused in the base period.
- Processor utilization factor with deadline is 1.00000 (see [1], page 6).
- Processor utilization factor with period is 1.00000 (see [1], page 6).
- In the preemptive case, with EDF, the task set is schedulable because the processor utilization factor 1.00000 is equal or less than 1.00000 (see [1], page 8,
theorem 2).

2) Feasibility  test based on worst case task response time :

- Bound on task response time :
    Task1 => 3
    Task2 => 6
    Task3 => 9
- All task deadlines will be met : the task set is schedulable.
```

Figure 27

LEAST LAXITY FIRST ANALYSIS:


Task name=Task1 Period= 3; Capacity= 1; Deadline= 3; Start time= 0; Priority= 1; Cpu=CPU0
Task name=Task2 Period= 6; Capacity= 2; Deadline= 6; Start time= 0; Priority= 2; Cpu=CPU0
Task name=Task3 Period= 9; Capacity= 3; Deadline= 9; Start time= 0; Priority= 3; Cpu=CPU0

```
Scheduling simulation, Processor CPU0 :
- Number of context switches :  15
- Number of preemptions :  5

- Task response time computed from simulation :
    Task1 => 1/worst
    Task2 => 5/worst
    Task3 => 9/worst
- No deadline missed in the computed scheduling : the task set seems to be schedulable.


Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The base period is 18 (see [18], page 5).
- 0 units of time are unused in the base period.
- Processor utilization factor with deadline is 1.00000 (see [1], page 6).
- Processor utilization factor with period is 1.00000 (see [1], page 6).
- In the preemptive case, with LLF, the task set is schedulable because the processor utilization factor 1.00000 is equal or less than 1.00000 (see [7]).

2) Feasibility  test based on worst case task response time :

- Bound on task response time :
    Task1 => 3
    Task2 => 6
    Task3 => 9
- All task deadlines will be met : the task set is schedulable.
```

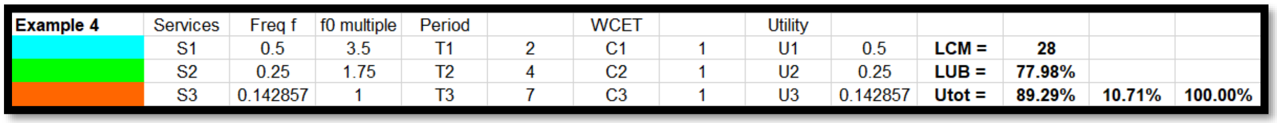
Figure 28

<u>EXPLANATION FOR SERVICE SET 6</u>: It can be observed that this task set fails to satisfy the completion time test and the scheduling point test. Thus, this task is not feasible. However, if this task set is scheduled using EDF or LLF policy, then this task set can be scheduled. The reason for this is similar to the case as explained for service set 1.

SERVICE SET 7

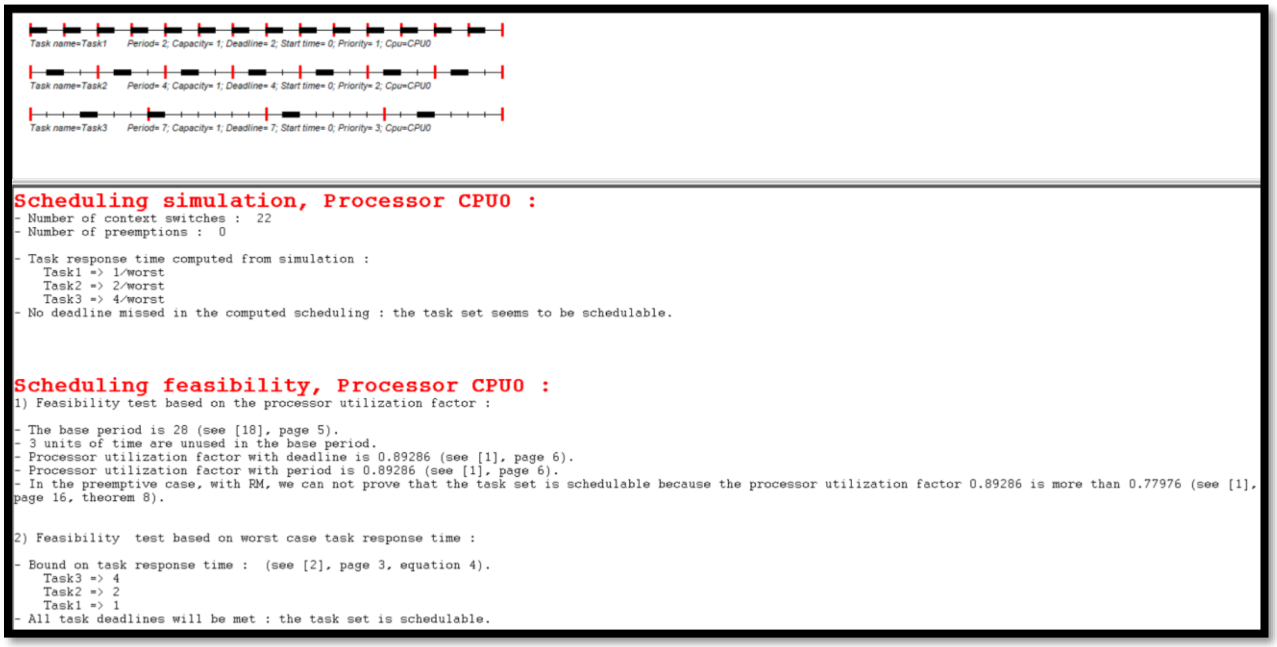
Time Periods: T1=2 T2=4 T3=7

Run-Times: C1=1 C2=1 C3=1



Example 4	Services	Freq f	f0 multiple	Period		WCET		Utility					
🟦	S1	0.5	3.5	T1	2	C1	1	U1	0.5	**LCM =**	**28**		
🟩	S2	0.25	1.75	T2	4	C2	1	U2	0.25	**LUB =**	**77.98%**		
🟧	S3	0.142857	1	T3	7	C3	1	U3	0.142857	**Utot =**	**89.29%**	**10.71%**	**100.00%**

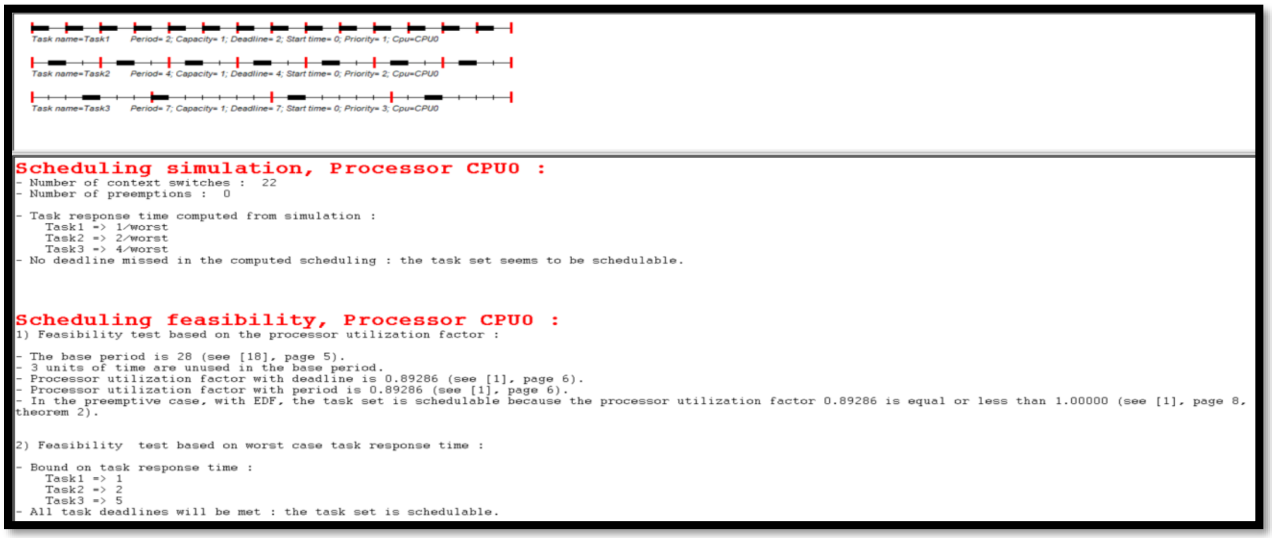
RATE MONOTONIC ANALYSIS:


Task name=Task1 Period= 2; Capacity= 1; Deadline= 2; Start time= 0; Priority= 1; Cpu=CPU0
Task name=Task2 Period= 4; Capacity= 1; Deadline= 4; Start time= 0; Priority= 2; Cpu=CPU0
Task name=Task3 Period= 7; Capacity= 1; Deadline= 7; Start time= 0; Priority= 3; Cpu=CPU0

```
Scheduling simulation, Processor CPU0 :
- Number of context switches :  22
- Number of preemptions :  0

- Task response time computed from simulation :
    Task1 => 1/worst
    Task2 => 2/worst
    Task3 => 4/worst
- No deadline missed in the computed scheduling : the task set seems to be schedulable.


Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The base period is 28 (see [18], page 5).
- 3 units of time are unused in the base period.
- Processor utilization factor with deadline is 0.89286 (see [1], page 6).
- Processor utilization factor with period is 0.89286 (see [1], page 6).
- In the preemptive case, with RM, we can not prove that the task set is schedulable because the processor utilization factor 0.89286 is more than 0.77976 (see [1],
page 16, theorem 8).


2) Feasibility  test based on worst case task response time :

- Bound on task response time :  (see [2], page 3, equation 4).
    Task3 => 4
    Task2 => 2
    Task1 => 1
- All task deadlines will be met : the task set is schedulable.
```

Figure 29

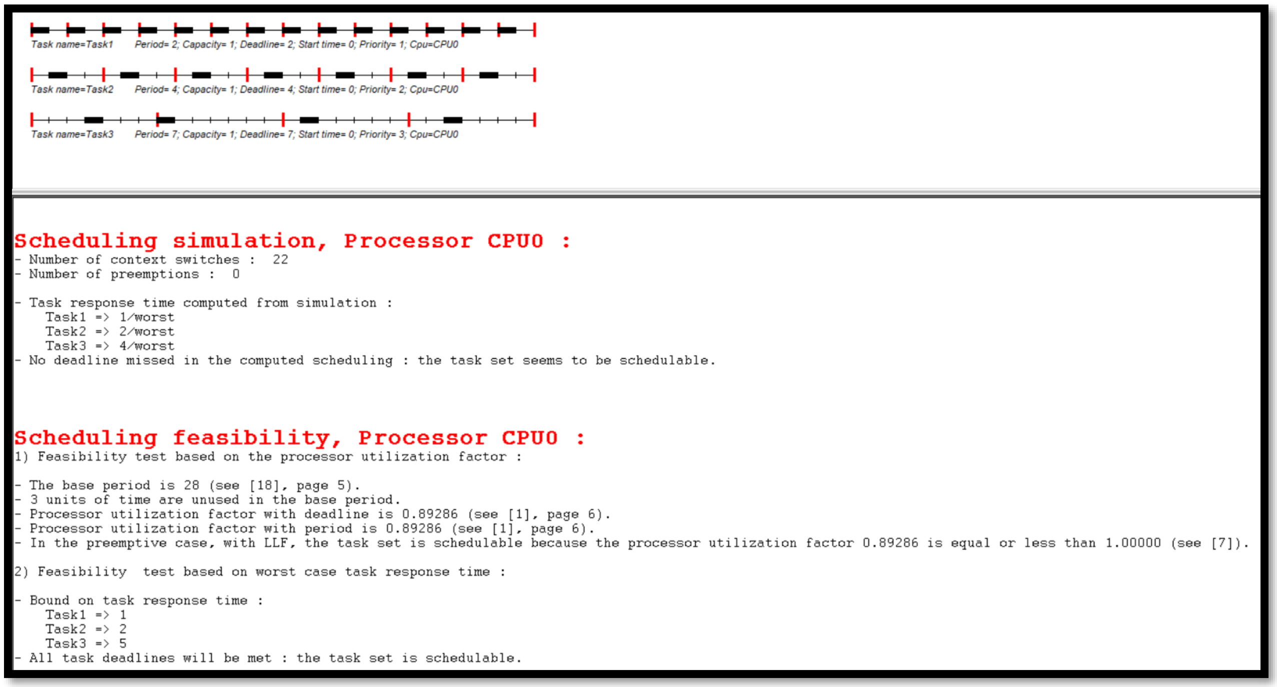
EARLIEST DEADLINE FIRST ANALYSIS:


Task name=Task1 Period= 2; Capacity= 1; Deadline= 2; Start time= 0; Priority= 1; Cpu=CPU0
Task name=Task2 Period= 4; Capacity= 1; Deadline= 4; Start time= 0; Priority= 2; Cpu=CPU0
Task name=Task3 Period= 7; Capacity= 1; Deadline= 7; Start time= 0; Priority= 3; Cpu=CPU0

```
Scheduling simulation, Processor CPU0 :
- Number of context switches :  22
- Number of preemptions :  0

- Task response time computed from simulation :
    Task1 => 1/worst
    Task2 => 2/worst
    Task3 => 4/worst
- No deadline missed in the computed scheduling : the task set seems to be schedulable.


Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The base period is 28 (see [18], page 5).
- 3 units of time are unused in the base period.
- Processor utilization factor with deadline is 0.89286 (see [1], page 6).
- Processor utilization factor with period is 0.89286 (see [1], page 6).
- In the preemptive case, with EDF, the task set is schedulable because the processor utilization factor 0.89286 is equal or less than 1.00000 (see [1], page 8,
theorem 2).


2) Feasibility  test based on worst case task response time :

- Bound on task response time :
    Task1 => 1
    Task2 => 2
    Task3 => 5
- All task deadlines will be met : the task set is schedulable.
```

Figure 30

LEAST LAXITY FIRST ANALYSIS:


Task name=Task1 Period= 2; Capacity= 1; Deadline= 2; Start time= 0; Priority= 1; Cpu=CPU0
Task name=Task2 Period= 4; Capacity= 1; Deadline= 4; Start time= 0; Priority= 2; Cpu=CPU0
Task name=Task3 Period= 7; Capacity= 1; Deadline= 7; Start time= 0; Priority= 3; Cpu=CPU0

```
Scheduling simulation, Processor CPU0 :
- Number of context switches :  22
- Number of preemptions :  0

- Task response time computed from simulation :
    Task1 => 1/worst
    Task2 => 2/worst
    Task3 => 4/worst
- No deadline missed in the computed scheduling : the task set seems to be schedulable.


Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The base period is 28 (see [18], page 5).
- 3 units of time are unused in the base period.
- Processor utilization factor with deadline is 0.89286 (see [1], page 6).
- Processor utilization factor with period is 0.89286 (see [1], page 6).
- In the preemptive case, with LLF, the task set is schedulable because the processor utilization factor 0.89286 is equal or less than 1.00000 (see [7]).

2) Feasibility  test based on worst case task response time :

- Bound on task response time :
    Task1 => 1
    Task2 => 2
    Task3 => 5
- All task deadlines will be met : the task set is schedulable.
```

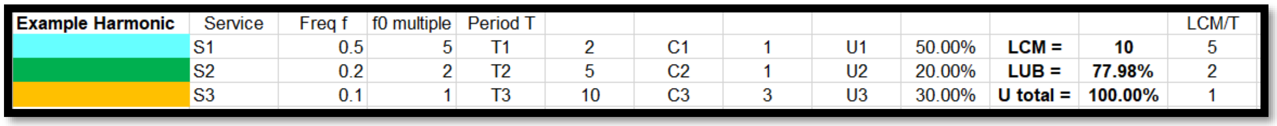
Figure 31

EXPLANATION FOR SERVICE SET 7: It can be observed that this task set is successful in passing the completion time test and the scheduling point test. Thus, this task is feasible. This task set can also be scheduled using EDF or LLF policy.

SERVICE SET 8

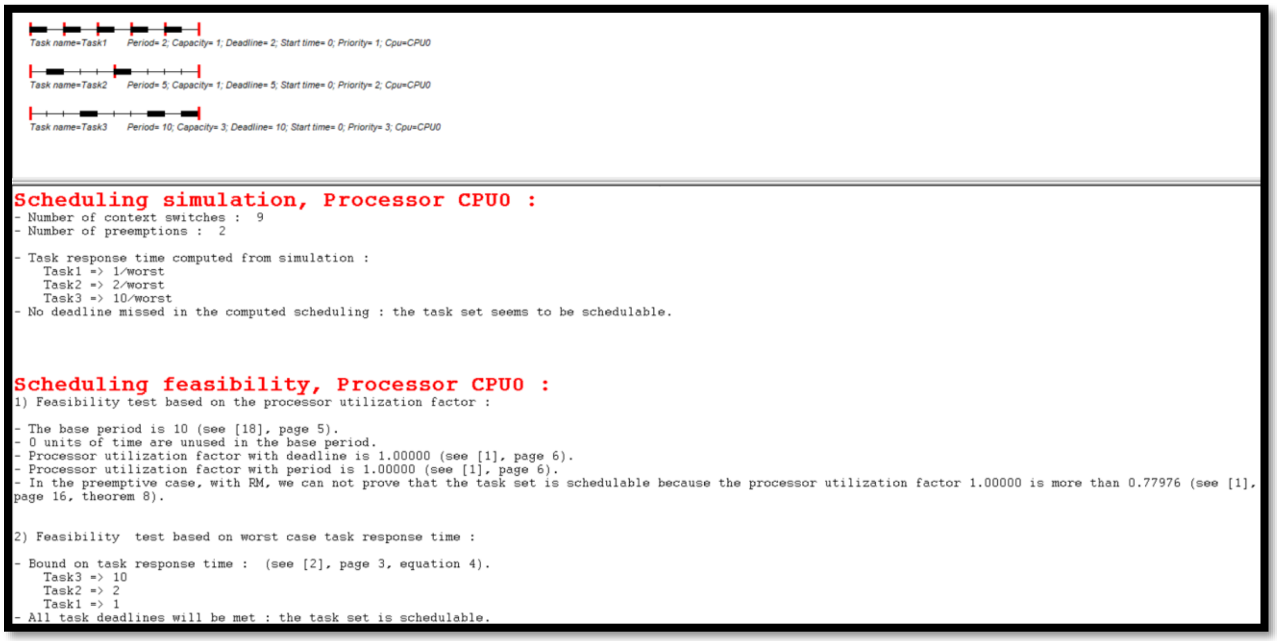
Time Periods: T1=2 T2=5 T3=10

Run-Times: C1=1 C2=1 C3=3



Example Harmonic	Service	Freq f	f0 multiple	Period T								LCM/T
🟦	S1	0.5	5	T1	2	C1	1	U1	50.00%	**LCM =**	**10**	5
🟩	S2	0.2	2	T2	5	C2	1	U2	20.00%	**LUB =**	**77.98%**	2
🟧	S3	0.1	1	T3	10	C3	3	U3	30.00%	**U total =**	**100.00%**	1

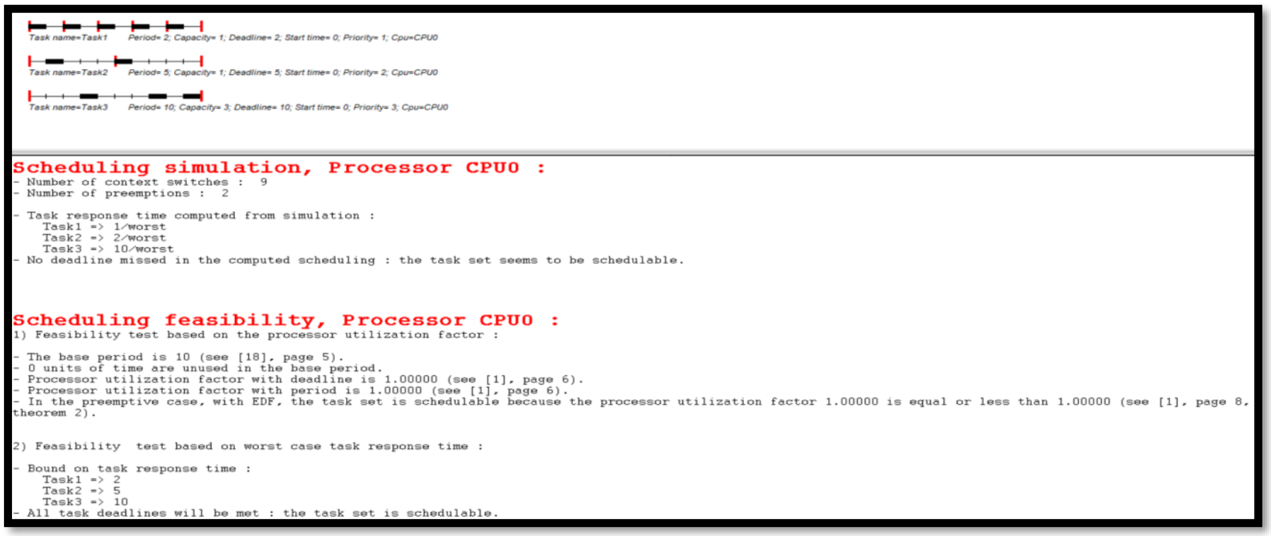
RATE MONOTONIC ANALYSIS:


Task name=Task1 Period= 2; Capacity= 1; Deadline= 2; Start time= 0; Priority= 1; Cpu=CPU0
Task name=Task2 Period= 5; Capacity= 1; Deadline= 5; Start time= 0; Priority= 2; Cpu=CPU0
Task name=Task3 Period= 10; Capacity= 3; Deadline= 10; Start time= 0; Priority= 3; Cpu=CPU0

```
Scheduling simulation, Processor CPU0 :
- Number of context switches :  9
- Number of preemptions :  2

- Task response time computed from simulation :
    Task1 => 1/worst
    Task2 => 2/worst
    Task3 => 10/worst
- No deadline missed in the computed scheduling : the task set seems to be schedulable.


Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The base period is 10 (see [18], page 5).
- 0 units of time are unused in the base period.
- Processor utilization factor with deadline is 1.00000 (see [1], page 6).
- Processor utilization factor with period is 1.00000 (see [1], page 6).
- In the preemptive case, with RM, we can not prove that the task set is schedulable because the processor utilization factor 1.00000 is more than 0.77976 (see [1],
page 16, theorem 8).


2) Feasibility  test based on worst case task response time :

- Bound on task response time :  (see [2], page 3, equation 4).
    Task3 => 10
    Task2 => 2
    Task1 => 1
- All task deadlines will be met : the task set is schedulable.
```

Figure 32

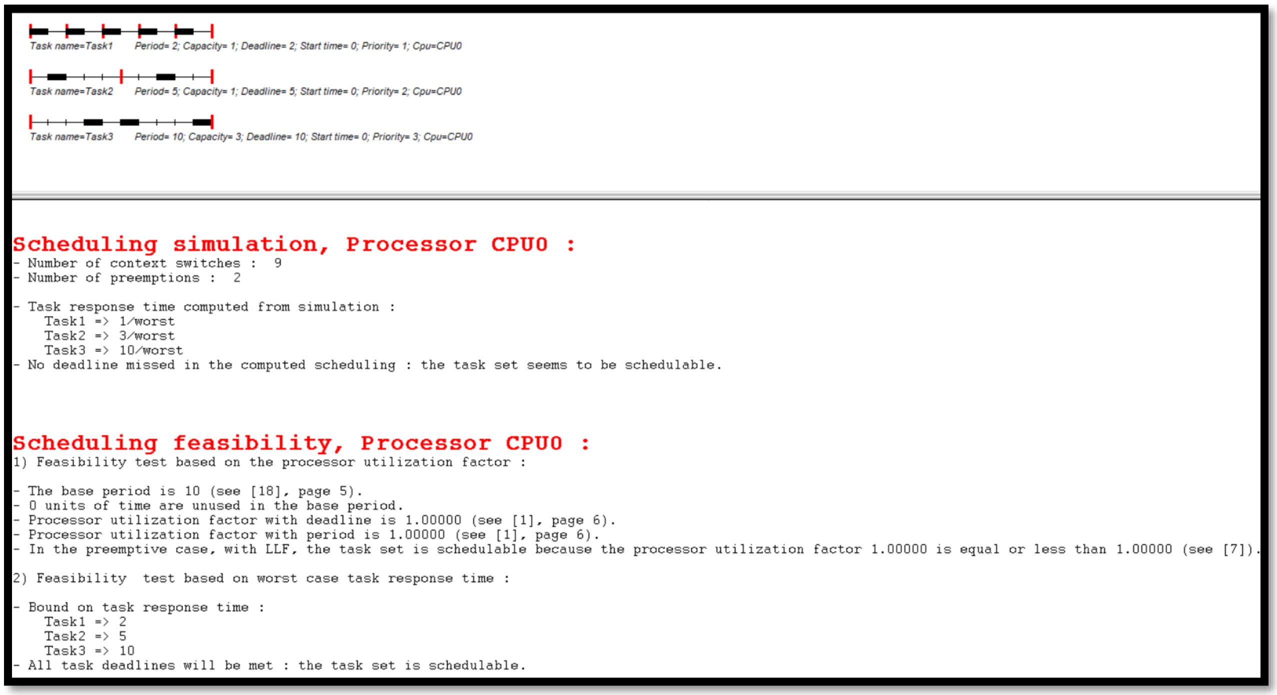
EARLIEST DEADLINE FIRST ANALYSIS:


Task name=Task1 Period= 2; Capacity= 1; Deadline= 2; Start time= 0; Priority= 1; Cpu=CPU0
Task name=Task2 Period= 5; Capacity= 1; Deadline= 5; Start time= 0; Priority= 2; Cpu=CPU0
Task name=Task3 Period= 10; Capacity= 3; Deadline= 10; Start time= 0; Priority= 3; Cpu=CPU0

```
Scheduling simulation, Processor CPU0 :
- Number of context switches :  9
- Number of preemptions :  2

- Task response time computed from simulation :
    Task1 => 1/worst
    Task2 => 2/worst
    Task3 => 10/worst
- No deadline missed in the computed scheduling : the task set seems to be schedulable.


Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The base period is 10 (see [18], page 5).
- 0 units of time are unused in the base period.
- Processor utilization factor with deadline is 1.00000 (see [1], page 6).
- Processor utilization factor with period is 1.00000 (see [1], page 6).
- In the preemptive case, with EDF, the task set is schedulable because the processor utilization factor 1.00000 is equal or less than 1.00000 (see [1], page 8,
theorem 2).


2) Feasibility  test based on worst case task response time :

- Bound on task response time :
    Task1 => 2
    Task2 => 5
    Task3 => 10
- All task deadlines will be met : the task set is schedulable.
```

Figure 33

LEAST LAXITY FIRST ANALYSIS:

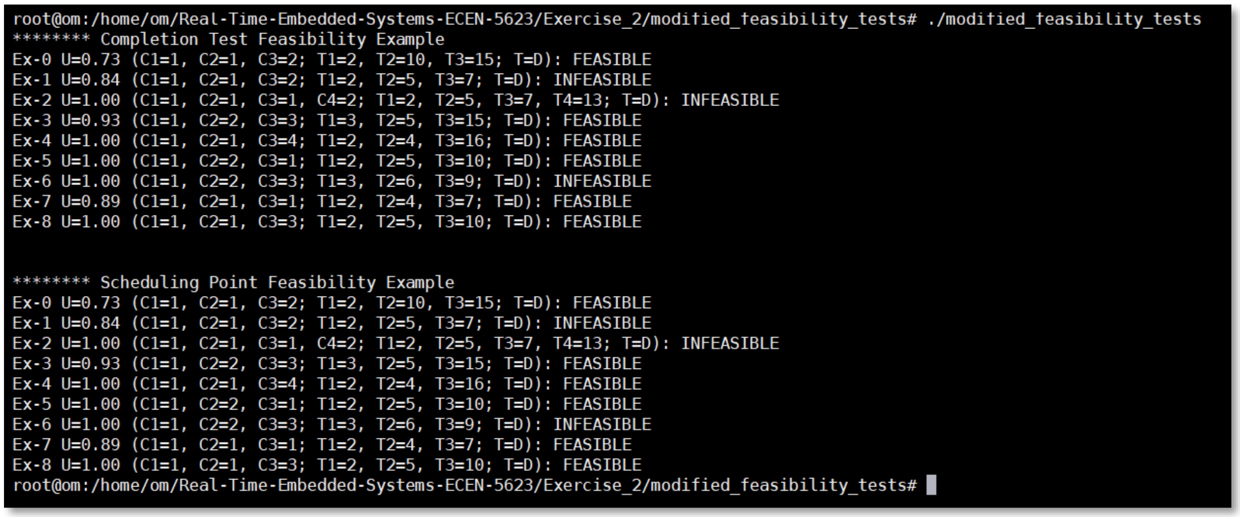

Task name=Task1 Period= 2; Capacity= 1; Deadline= 2; Start time= 0; Priority= 1; Cpu=CPU0
Task name=Task2 Period= 5; Capacity= 1; Deadline= 5; Start time= 0; Priority= 2; Cpu=CPU0
Task name=Task3 Period= 10; Capacity= 3; Deadline= 10; Start time= 0; Priority= 3; Cpu=CPU0

```
Scheduling simulation, Processor CPU0 :
- Number of context switches :  9
- Number of preemptions :  2

- Task response time computed from simulation :
    Task1 => 1/worst
    Task2 => 3/worst
    Task3 => 10/worst
- No deadline missed in the computed scheduling : the task set seems to be schedulable.


Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The base period is 10 (see [18], page 5).
- 0 units of time are unused in the base period.
- Processor utilization factor with deadline is 1.00000 (see [1], page 6).
- Processor utilization factor with period is 1.00000 (see [1], page 6).
- In the preemptive case, with LLF, the task set is schedulable because the processor utilization factor 1.00000 is equal or less than 1.00000 (see [7]).

2) Feasibility  test based on worst case task response time :

- Bound on task response time :
    Task1 => 2
    Task2 => 5
    Task3 => 10
- All task deadlines will be met : the task set is schedulable.
```

Figure 34

EXPLANATION FOR SERVICE SET 8: It can be observed that this task set is successful in passing the completion time test and the scheduling point test. Thus, this task is feasible. This task set can also be scheduled using EDF or LLF policy.

Test Results for Modified Feasibility Example Code:



```
root@om:/home/om/Real-Time-Embedded-Systems-ECEN-5623/Exercise_2/modified_feasibility_tests# ./modified_feasibility_tests
******** Completion Test Feasibility Example
Ex-0 U=0.73 (C1=1, C2=1, C3=2; T1=2, T2=10, T3=15; T=D): FEASIBLE
Ex-1 U=0.84 (C1=1, C2=1, C3=2; T1=2, T2=5, T3=7; T=D): INFEASIBLE
Ex-2 U=1.00 (C1=1, C2=1, C3=1, C4=2; T1=2, T2=5, T3=7, T4=13; T=D): INFEASIBLE
Ex-3 U=0.93 (C1=1, C2=2, C3=3; T1=3, T2=5, T3=15; T=D): FEASIBLE
Ex-4 U=1.00 (C1=1, C2=1, C3=4; T1=2, T2=4, T3=16; T=D): FEASIBLE
Ex-5 U=1.00 (C1=1, C2=2, C3=1; T1=2, T2=5, T3=10; T=D): FEASIBLE
Ex-6 U=1.00 (C1=1, C2=2, C3=3; T1=3, T2=6, T3=9; T=D): INFEASIBLE
Ex-7 U=0.89 (C1=1, C2=1, C3=1; T1=2, T2=4, T3=7; T=D): FEASIBLE
Ex-8 U=1.00 (C1=1, C2=1, C3=3; T1=2, T2=5, T3=10; T=D): FEASIBLE


******** Scheduling Point Feasibility Example
Ex-0 U=0.73 (C1=1, C2=1, C3=2; T1=2, T2=10, T3=15; T=D): FEASIBLE
Ex-1 U=0.84 (C1=1, C2=1, C3=2; T1=2, T2=5, T3=7; T=D): INFEASIBLE
Ex-2 U=1.00 (C1=1, C2=1, C3=1, C4=2; T1=2, T2=5, T3=7, T4=13; T=D): INFEASIBLE
Ex-3 U=0.93 (C1=1, C2=2, C3=3; T1=3, T2=5, T3=15; T=D): FEASIBLE
Ex-4 U=1.00 (C1=1, C2=1, C3=4; T1=2, T2=4, T3=16; T=D): FEASIBLE
Ex-5 U=1.00 (C1=1, C2=2, C3=1; T1=2, T2=5, T3=10; T=D): FEASIBLE
Ex-6 U=1.00 (C1=1, C2=2, C3=3; T1=3, T2=6, T3=9; T=D): INFEASIBLE
Ex-7 U=0.89 (C1=1, C2=1, C3=1; T1=2, T2=4, T3=7; T=D): FEASIBLE
Ex-8 U=1.00 (C1=1, C2=1, C3=3; T1=2, T2=5, T3=10; T=D): FEASIBLE
root@om:/home/om/Real-Time-Embedded-Systems-ECEN-5623/Exercise_2/modified_feasibility_tests#
```

Figure 35

4. Provide 3 constraints that are made on the RM LUB derivation and 3 assumptions as documented in the Liu and Layland paper and in Chapter 3 of RTECS with Linux and RTOS. Finally, list 3 key derivation steps in the RM LUB derivation that you either do not understand or that you would consider "tricky" math. Attempt to describe the rationale for those steps as best you can do based upon reading in Chapter 3 of RTECS with Linux and RTOS.

Ans: <u>Three assumptions documented in the Liu Layland paper:</u>

1) Requests for all tasks for which hard deadlines exist are periodic, with constant interval between requests.
2) The tasks are independent in that requests for a certain task do not depend on the initiation or the completion of requests for other tasks.
3) Run-time for each task is constant for that task and does not vary with time.

<u>Three constraints made on RM LUB derivation:</u>

1) The deadline for a task is equal to its time period. This means that each task should provide response before the next time it is about to occur. This makes the derivation a lot simpler.
2) The time taken by the CPU to switch between lower and higher priority task has not been considered. This is because the time taken to context switch can vary and hence can make the derivation of the RM LUB more complex. Thus, to avoid this complexity, the time taken in context switching between multiple tasks has not been taken into account.
3) For the assumption (Each task must be completed before a next request for it occurs) to hold true, a small but significant amount of buffering hardware must exist for each peripheral function.

<u>Three Key derivation Steps:</u>

1) After reading the Lui Layland paper, I was not able to understand how exactly the authors arrived at the equations of C_1 and C_2. After referring to the RTES book, I could understand how C_1 and C_2 arrived.
   The first case of the RM LUB derivation states that C_1 is short enough to fit all three releases in T_2. This derivation made more sense when this was explained using the following diagram.

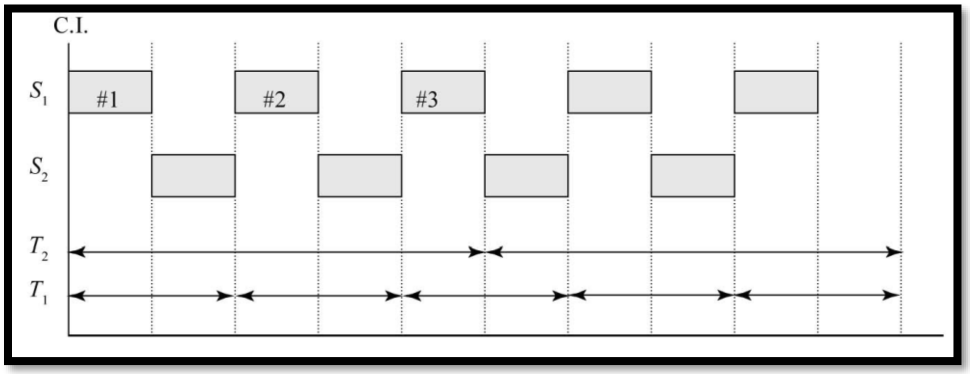


Figure 36

In this case, all the three S_1 releases that require C_1 execution time fit in time period T_2. The equation of C_1 is,

$$\boxed{C_1 \le T_2 - T_1 \lfloor T_2 / T_1 \rfloor}$$

Based on C_1, C_2 was also calculated using its time period and the interference from C_1.

2) The one statement in the proof says that "The smallest *I* possible is 1, and the LUB for U occurs when I is minimized, so we substitute 1 for I ". Here, I was not able to understand how exactly putting value of I as 1 will affect the LUB.

3) In the Liu Layland paper, it is stated that U monotonically decreases in C_1. But after reading through the proof provided in the RTECS book, the explanation provided to prove that U decreases monotonically with increase in C_1 helped clear that doubt. The graphical plot shown below provides a visual context to the statement made in the Liu Layland paper. The manner in which the explanation is broken down helps provide a clear understanding of the statement "U decreases monotonically with increase in C_1".

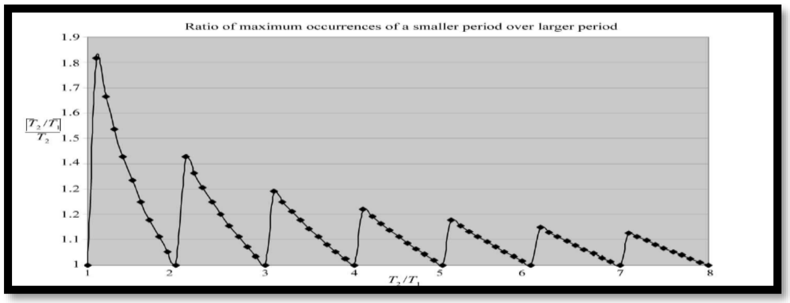


Figure 37

TABLE OF FIGURES:

REFERENCES:

[1] Liu Layland paper http://mercury.pr.erau.edu/~siewerts/cec450/documents/Papers/liu_layland.pdf

[2] Real-time embedded components and systems with Linux and RTOS by *Sam Siewert* and *John Pratt*.

[3] Architecture of the space shuttle primary avionics software system by *Gene Carlow.*

[4] Cheddar software for scheduling analysis.

[5] GitHub Link : https://github.com/omraheja/Real-Time-Embedded-Systems-ECEN-5623