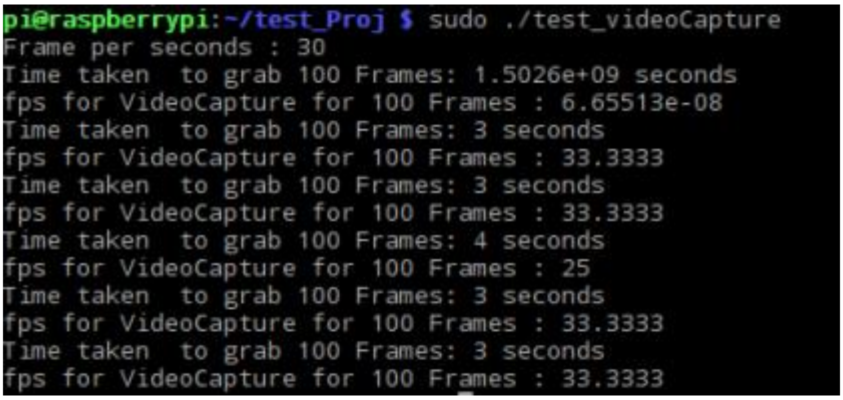
**RM LUB: Project #9**

Since we are assuming RT scheduling, the T=D

We have 5 services.

1. Video Capture
2. Sequencer
3. Warning System
4. Object Detection
5. Lane Detection

## 1. Video Capture:



```
pi@raspberrypi:~/test_Proj $ sudo ./test_videoCapture
Frame per seconds : 30
Time taken  to grab 100 Frames: 1.5026e+09 seconds
fps for VideoCapture for 100 Frames : 6.65513e-08
Time taken  to grab 100 Frames: 3 seconds
fps for VideoCapture for 100 Frames : 33.3333
Time taken  to grab 100 Frames: 3 seconds
fps for VideoCapture for 100 Frames : 33.3333
Time taken  to grab 100 Frames: 4 seconds
fps for VideoCapture for 100 Frames : 25
Time taken  to grab 100 Frames: 3 seconds
fps for VideoCapture for 100 Frames : 33.3333
Time taken  to grab 100 Frames: 3 seconds
fps for VideoCapture for 100 Frames : 33.3333
```

The Frames per second for the Video Capture is 30 => by default.  (as per the specs)
With Bright ambient conditions the FPS => 25
With least ambient conditions the FPS => 33

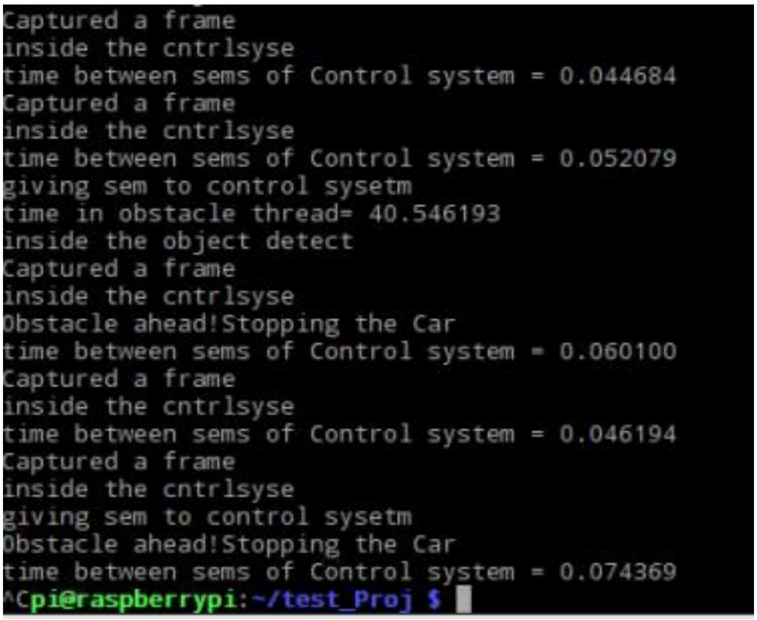
Considering the WCET, we have assumed the C1 to be 33ms ; Thus we choose the T to be 70ms (with 37ms for buffer). Thus if T=70ms => the frequency of the First service = 28Hz

C1 = 33ms

T1 = 70ms

U1= C1/T1 = 0.471 =>  47.1% CPU is utilised

## 3. Warning System



```
Captured a frame
inside the cntrlsyse
time between sems of Control system = 0.044684
Captured a frame
inside the cntrlsyse
time between sems of Control system = 0.052079
giving sem to control sysetm
time in obstacle thread= 40.546193
inside the object detect
Captured a frame
inside the cntrlsyse
Obstacle ahead!Stopping the Car
time between sems of Control system = 0.060100
Captured a frame
inside the cntrlsyse
time between sems of Control system = 0.046194
Captured a frame
inside the cntrlsyse
giving sem to control sysetm
Obstacle ahead!Stopping the Car
time between sems of Control system = 0.074369
^Cpi@raspberrypi:~/test_Proj $
```

C3 = 1ms (it just has conditional logic statements) [it is actually 0.094ms = 94us; but assuming that the worst case time will be atleast 1ms]

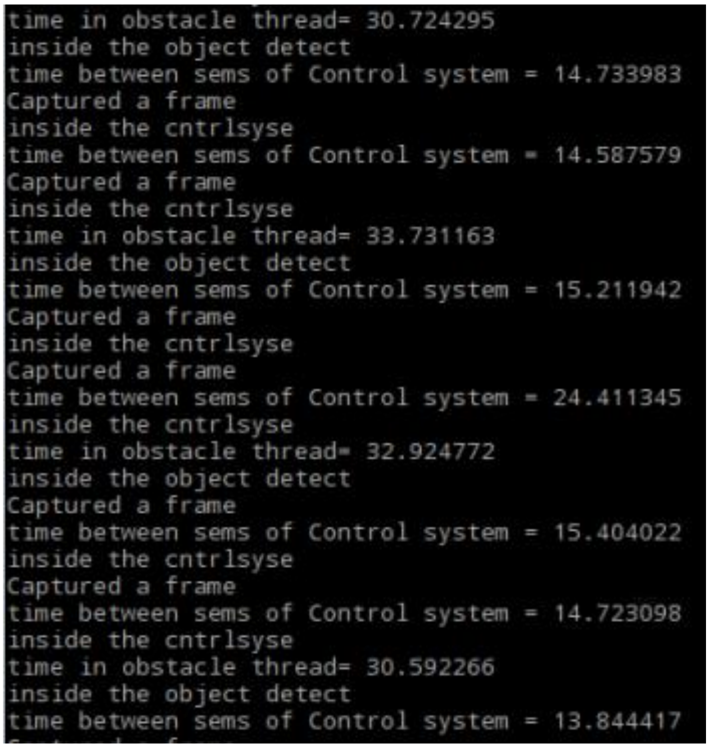
According to our real-time application flow, the Warning system should be updated with warnings from Obstacle Detection and Lane Detection.
To make the frequency of the Warning System appropriate to the Obstacle & Lane detection system, We have chosen the T (The Deadline) to be =>

T3 = 10ms

There for making the CPU Utilization factor to be:
C3 = 1ms

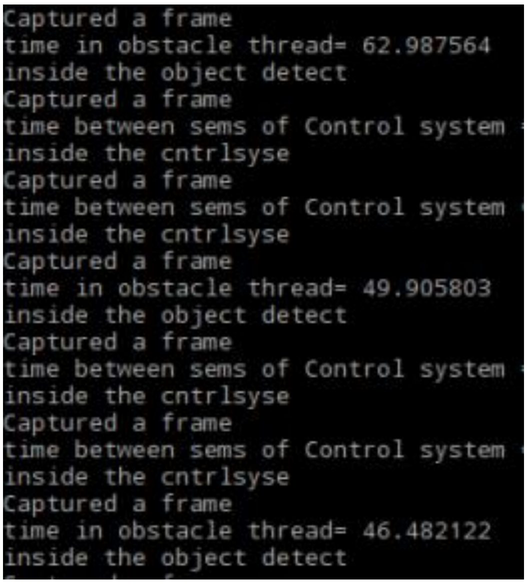
T3 = 10ms

U3= C3/T3 = 0.1 => 10% of CPU is utilized.



```
time in obstacle thread= 30.724295
inside the object detect
time between sems of Control system = 14.733983
Captured a frame
inside the cntrlsyse
time between sems of Control system = 14.587579
Captured a frame
inside the cntrlsyse
time in obstacle thread= 33.731163
inside the object detect
time between sems of Control system = 15.211942
Captured a frame
inside the cntrlsyse
Captured a frame
time between sems of Control system = 24.411345
inside the cntrlsyse
time in obstacle thread= 32.924772
inside the object detect
Captured a frame
time between sems of Control system = 15.404022
inside the cntrlsyse
Captured a frame
time between sems of Control system = 14.723098
inside the cntrlsyse
time in obstacle thread= 30.592266
inside the object detect
time between sems of Control system = 13.844417
```

## 4. Obstacle Detection



```
Captured a frame
time in obstacle thread= 62.987564
inside the object detect
Captured a frame
time between sems of Control system
inside the cntrlsyse
Captured a frame
time between sems of Control system
inside the cntrlsyse
Captured a frame
time in obstacle thread= 49.905803
inside the object detect
Captured a frame
time between sems of Control system
inside the cntrlsyse
Captured a frame
time between sems of Control system
inside the cntrlsyse
Captured a frame
time in obstacle thread= 46.482122
inside the object detect
```

C4 = 60ms (Worst Execution Time)

T4 = 100ms (to leave some room for buffer and not utilize the entire CPU)

U4 = C4/T4 = 0.6 => 60 % of CPU is utilized.

After jitter analysis for different videos, we found that the –ve jitter was crossing more than 5% of the deadline. Thus we changed the value of T4 to 150ms

Thus U4 = C4/T4 = 60/150 = 40% of CPU

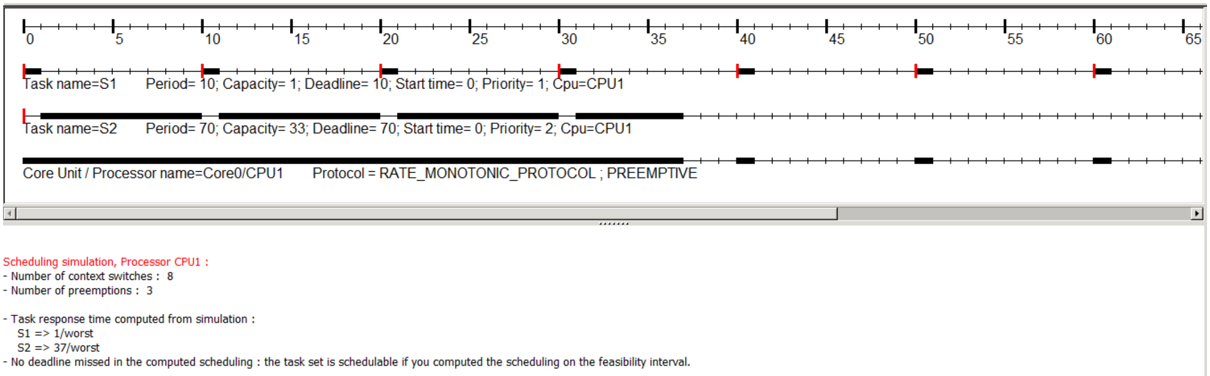
**5. Lane Detection**

C5 = 500ms (Worst Case Execution Time)

T5 = 650ms (to leave some room for buffer and not utilize the entire CPU)

U5 = C5/T5 = 0.76 => 76.92% of the CPU will be utilised.

Cheddar analysis for Video Capture and the control system in same Core:



```
Task name=S1      Period= 10; Capacity= 1; Deadline= 10; Start time= 0; Priority= 1; Cpu=CPU1
Task name=S2      Period= 70; Capacity= 33; Deadline= 70; Start time= 0; Priority= 2; Cpu=CPU1
Core Unit / Processor name=Core0/CPU1      Protocol = RATE_MONOTONIC_PROTOCOL ; PREEMPTIVE

Scheduling simulation, Processor CPU1 :
- Number of context switches :  8
- Number of preemptions :  3

- Task response time computed from simulation :
   S1 => 1/worst
   S2 => 37/worst
- No deadline missed in the computed scheduling : the task set is schedulable if you computed the scheduling on the feasibility interval.
```

S1 and S3 are put in same core since the utilization factors macth and its less than 100% making it have a less margin.
so

C1/T1 = 0.471 => 47%

C3/T3 = 0.1 => 10 %

Totally 57% which is in the bounds of RM LUB 78% margin.

| Service | C (ms) | T (ms) | U (CPU %) | Frequency (Hz) | Priority |
|---|---|---|---|---|---|
| Lane Detection | 500 | 650 | 76.92 | 1.54 | 4 |
| Obstacle Detection | 60 | 150 | 40% | 6.66 | 5 |
| Video Capture | 33 | 70 | 47.14% | 14.28 | 3 |
| Warning System | 1 | 10 | 10% | 100hz | 2 |
| Sequencer | 10 | 10 | 100% (But practically this will run at less than that) | 100hz | 1 |