Implementing EDF scheduler

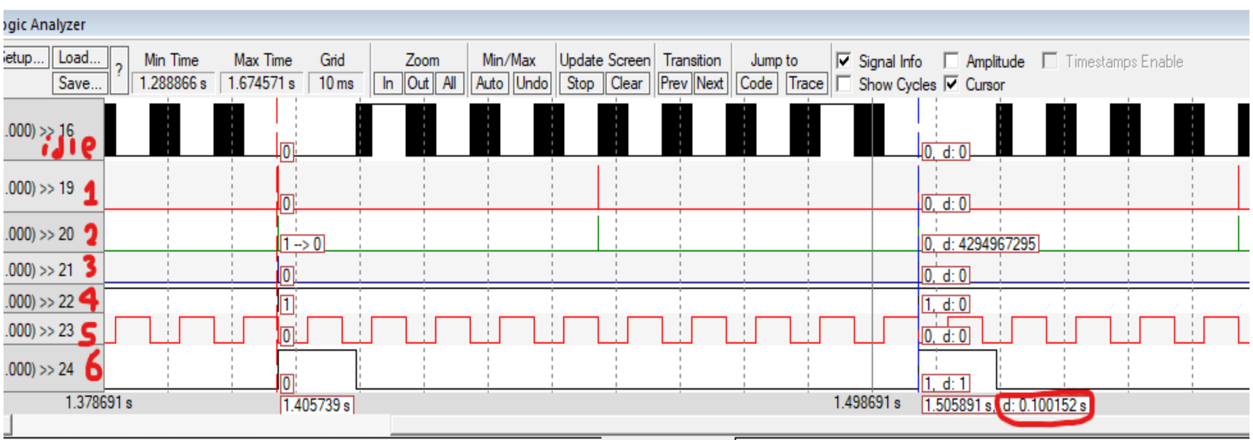
Verifying the implementation:

1- Analytical methods

Calculating the system hyperperiod:

After implementing the 6 provided tasks on Keil, GPIOs and logic analyzer were used to track when tasks are running, so we would be able to determine the hyperperiod.

Below is a screenshot of the 6 tasks + the idle task, we can easily detect a pattern that is repeated every 100ms, **so the hyperperiod is 100ms.**

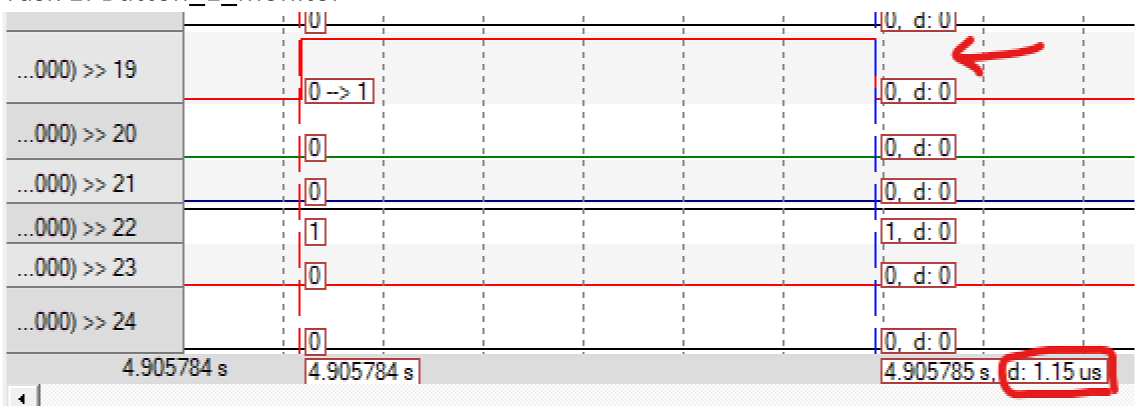


Calculating the CPU load:

The CPU load is calculated by dividing the execution time of every task by its period, then adding them all together.

But there are tasks with no execution time provided, so we can calculate it using GPIOs and logic analyzer.

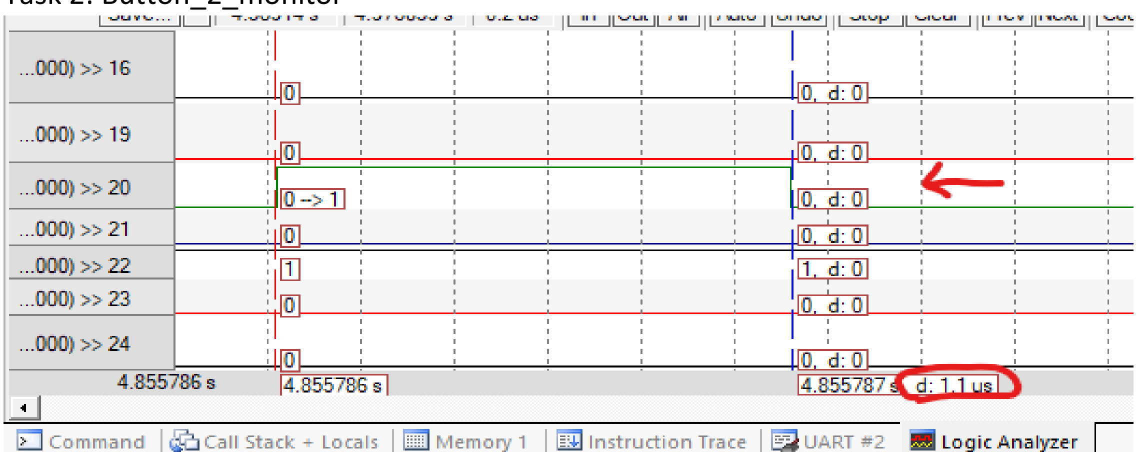
Task 1: Button_1_monitor



We can clearly see that the execution time for this task is 1.15us, P=50ms

L1 = 1.15us/50ms = 0.000023

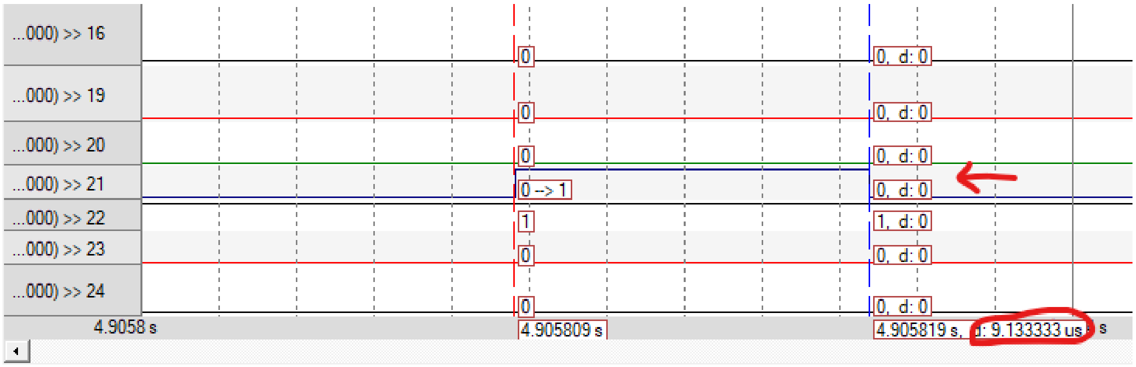
Task 2: Button_2_monitor



We can clearly see that the execution time for this task is 1.1us, P=50ms

L1 = 1.1us/50ms = 0.000022

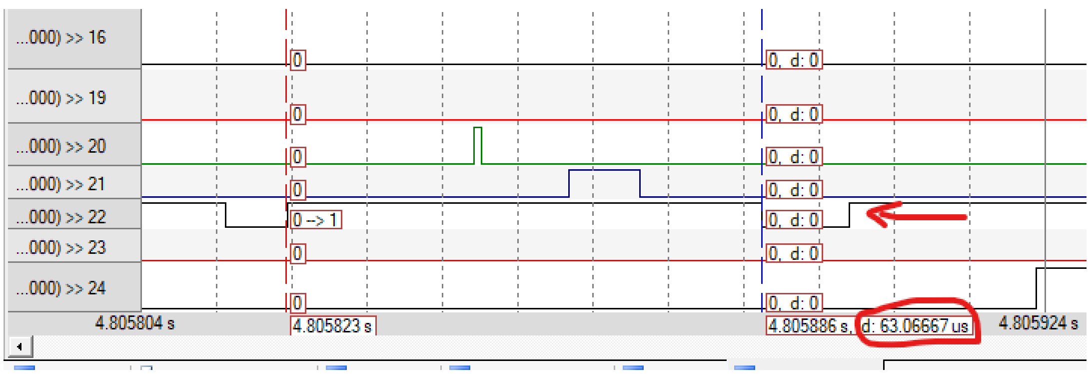
Task 3: Periodic transmitter



We can clearly see that the execution time for this task is 9.13us, P=100ms

L1 = 9.13us/100ms = 0.0000913

Task 4: UART Receiver



We can clearly see that the execution time for this task is 63us , P=20ms

L1 = 63us/20ms = 0.00315

Task 5: Load 1

Given execution time for this task is 5ms , P=10ms

L1 = 5ms/10ms = 0.5

Task 6: Load 2

Given execution time for this task is 12ms , P=100ms

L1 = 12ms/100ms = 0.12

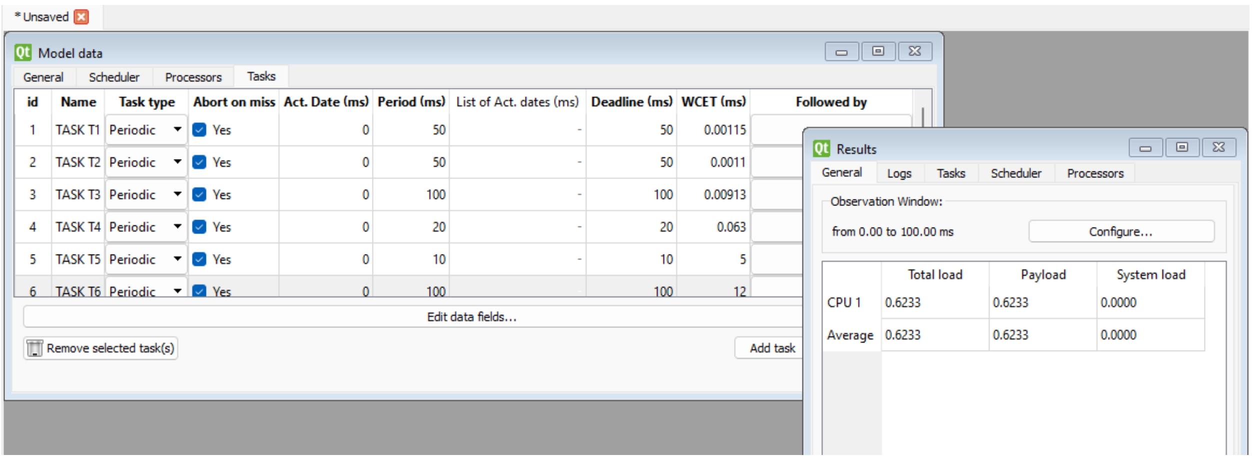
CPU load = (0.5 + 0.12 + 0.00315 + 0.0000913 + 0.000022 + 0.000023)% = 62.32%

Checking system schedulability using URM and time demand analysis techniques (Assuming the given set of tasks are scheduled using a fixed priority rate -monotonic scheduler):

Using URM method:

We test If U <= n*(2^(1/n)-1), where U is the CPU load, n is the number of tasks
I have applied all tasks on simso with the previous calculated execution times, and it resulted in a CPU load of 0.6233

so U = 0.6233, n = 6, n*(2^(1/n)-1) = 6*(2^(1/6)-1) = 0.73477
so U < n*(2^(1/n)-1). So U < URM
so the system is schedulable.



Using Time demand analysis:

Task 1: {P: 50, E: 0.00115, D:50} Priority: 4
Task 2: {P: 50, E: 0.0011, D:50} Priority: 3
Task 3: {P: 100, E: 0.00913, D:100} Priority: 2
Task 4: {P: 20, E: 0.063, D:20} Priority: 5
Task 5: {P: 10, E: 5, D:10} Priority: 6
Task 6: {P: 100, E: 12, D:100} Priority: 1

Task 5:

W(1) = 5 + 0 = 5
W(2) = 5 + 0 = 5
W(3) = 5 + 0 = 5
W(4) = 5 + 0 = 5
W(5) = 5 + 0 = 5 w(5)<T5 (10) so task 5 is feasible

Task 4:

W(1) = 0.063 + (1/10)*5 = 5.063
W(2) = 0.063 + (2/10)*5 = 5.063
W(3) = 0.063 + (3/10)*5 = 5.063
W(4) = 0.063 + (4/10)*5 = 5.063
W(5) = 0.063 + (5/10)*5 = 5.063
W(6) = 0.063 + (6/10)*5 = 5.063
W(7) = 0.063 + (7/10)*5 = 5.063
W(8) = 0.063 + (8/10)*5 = 5.063
W(9) = 0.063 + (9/10)*5 = 5.063
W(10) = 0.063 + (10/10)*5 = 5.063
W(11) = 0.063 + (11/10)*5 = 10.063
W(12) = 0.063 + (12/10)*5 = 10.063
W(13) = 0.063 + (13/10)*5 = 10.063
W(14) = 0.063 + (14/10)*5 = 10.063
W(15) = 0.063 + (15/10)*5 = 10.063
W(16) = 0.063 + (16/10)*5 = 10.063
W(17) = 0.063 + (17/10)*5 = 10.063
W(18) = 0.063 + (18/10)*5 = 10.063
W(19) = 0.063 + (19/10)*5 = 10.063
W(20) = 0.063 + (20/10)*5 = 10.063

w(20)<T4 (20) so task 4 is feasible

Task 1:

W(1) = 0. 00115 + (1/10)*5 + (1/20)*0.063 = 5.06415
W(11) = 0. 00115 + (11/10)*5 + (11/20)*0.063 = 10.06415
W(21) = 0. 00115 + (21/10)*5 + (21/20)*0.063 = 10.1275
…
W(50) = 0. 00115 + (50/10)*5 + (50/20)*0.063 = 25.19015

w(50)<T1 (50) so task 1 is feasible

Task 2:

W(1) = 0. 0011+ (1/10)*5 + (1/20)*0.063 + (1/50)*0.0015= 5.0656
…
W(50) = 0. 0011+ (50/10)*5 + (50/20)*0.063 + (50/50)*0.0015 = 25.19016

w(50)<T2 (50) so task 2 is feasible

Task 3:

W(1) = 0. 00913+ (1/10)*5 + (1/20)*0.063 + (1/50)*0.0015 + (1/50)*0.0011= 5.07473

…

W(100) = 0. 00913+ (100/10)*5 + (100/20)*0.063 + (100/50)*0.0015 + (100/50)*0.0011 = 50.32933

w(100)<T3 (100) so task 3 is feasible

Task 6:

W(1) = 12+ (1/10)*5 + (1/20)*0.063 + (1/50)*0.0015 + (1/50)*0.0011 + (1/100)*0. 00913 = 17.07473

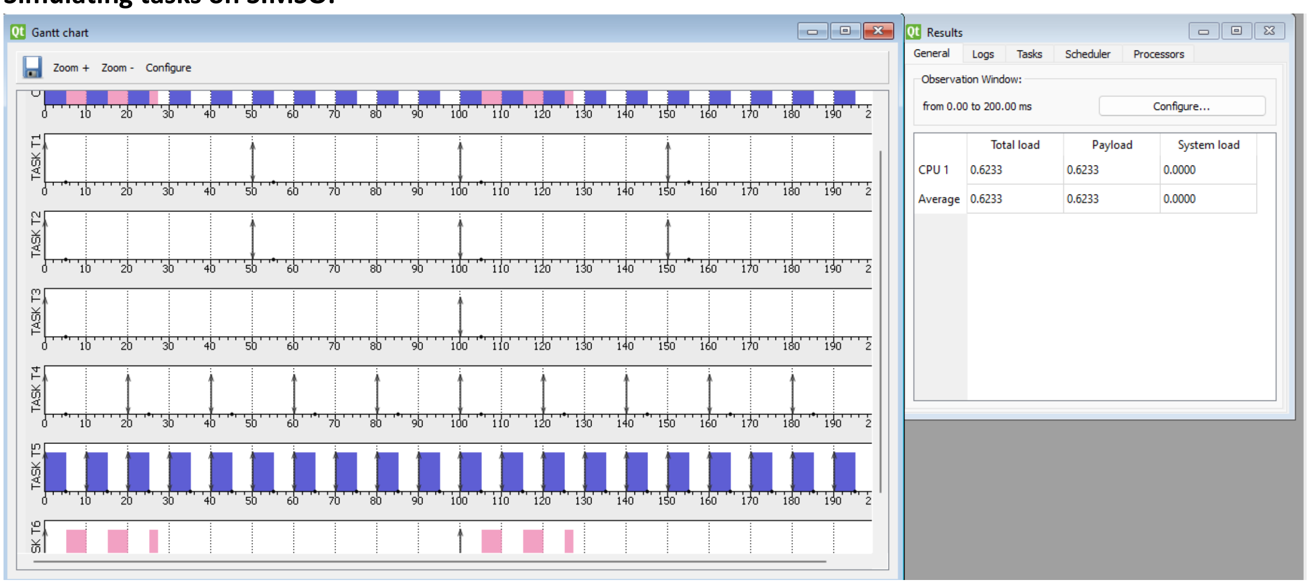
…

W(100) = 12+ (100/10)*5 + (100/20)*0.063 + (100/50)*0.0015 + (100/50)*0.0011 + (100/100) * 0. 00913 = 62.32933

w(100)<T6 (100) so task 6 is feasible

since all tasks are feasible, so the system is schedulable.

Simulating tasks on SIMSO:



3. Using Keil simulator in run-time and the given set of tasks: using rate monotonic