Real-Time Operating System Project
EDF Implementation Report

Name: **Ahmed Mohamed Hussein Elshehry**

Email : elshehry97@gmail.com

Verifying the System Implementation:

Method 1: Analytically

1. Calculate Hyper-period
2. CPU Load
3. Schedulability Analysis (using: A: Urm, B: Time Demand Analysis)

- using analytical methods:

1- System Hyper-period :

Task	Periodicity
Button 1 Monitor	50
Button 2 Monitor	50
Periodic Transmitter	100
UART Transmitter	20
Load 1 Simulation	10
Load 2 Simulation	100

$Hyperperiod$ = Least Common Multiplier of all tasks periodicities
$Hyperperiod = LCM$ (50, 50, 100, 20, 10, 100)
$Hyperperiod = 100$

2- CPU Load:

Task	Execution Time	Occurrence During Hyper-period
Button 1 Monitor	18.5 µs	2
Button 2 Monitor	19 µs	2
Periodic Transmitter	19.2 µs	1
UART Transmitter	21.6 µs	5
Load 1 Simulation	5 ms	10
Load 2 Simulation	12 ms	1

Note: Execution times of tasks calculated from the logic analyzer in Keil

Utilization = Total Execution Time During Hyper-period / Hyper-period

$$U = ((T1*2)+(T2*2)+T3*1)+(T4*5)+T5*10)+(T6*1) \ / \ 100m \ ) * 100\%$$

$$U = ((18{,}5\mu s*2)+(19\mu s*2)+(19.2\mu s*1)+(21{,}6\mu s*5)+(5ms*10)+(12ms*1) \ / \ 100ms \ )* 100\% = 62.22\%$$

3- System Schedulability:

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

- **<u>Using Rate Monotonic Utilization Bound</u>**

$$U \leq n[\ 2\ \hat{}(1/n)\ - 1]$$

Urm =n (2^(1/n) – 1) = 6(2^(1/6) – 1) = 0.73477

$U = \sum Ci/Pi = 18.5\ \mu s / 50\ ms + 19\ \mu s / 50\ ms + 19.2\ \mu s / 100\ ms + 21.6\ \mu s/20\ ms + 5\ ms /10\ ms + 12\ ms / 100\ ms = 0.62202$

Since U < Urm**, Therefore System guaranteed Schedulable**

- **Using Time Demand Analysis:**

$$w_i(t) = e_i + \sum_{k=1}^{i-1} \left\lceil \frac{t}{p_k} \right\rceil e_k \quad for\ 0 < t \le p_i$$

W = Worst response time
E = Execution time
P = Periodicity
T = Time instance

Arrange Tasks according to Priority will be:

[Task 5, Task 4, Task 1, Task 2, Task 3, Task 6]

Task	Task ID	Periodicity	Execution Time
Button 1 Monitor	Task 1	50	18.5 µs
Button 2 Monitor	Task 2	50	19 µs
Periodic Transmitter	Task 3	100	19.2 µs
UART Transmitter	Task 4	20	21.6 µs
Load 1 Simulation	Task 5	10	5 ms
Load 2 Simulation	Task 6	100	12 ms

For Task 5 (Load 1 Simulation)

W(1)……W(10)

W(1) = 5 + 0 = 5 ms → W(10) = 5 + 0 = 5 ms

W(10) < D = 5ms < 10 ms , **Task 5 is Schedulable**

For Task 4 (UART Transmitter)

W(1)…….W(20)
W(1) = 21.6 µs + (1/10)*5 ms = 0.522 ms
W(5) = 21.6 µs + (5/10)*5 ms = 2.5216 ms
W(10) = 21.6 µs + (10/10)*5 ms = 5.023 ms
W(20) = 21.6 µs + (20/10)*5 ms = 10.021 ms
W(20) < D = 10.021 ms < 20 ms , **Task 4 is Schedulable**

For Task 1 (Button 1 Monitor)

W(1)……W(50)
W(1) =18.5 µs + (1/10)*5 ms+ (1/20)* 21.6 µs ms = 0.519 ms
W(50) = 18.5 µs + (50/10)*5 ms+ (50/20)* 21.6 µs ms = 25.0725 ms

W(50) < D = 25.0725 ms <50ms , **Task 1 is Schedulable**

For Task 2 (Button 2 Monitor)

W(1)……W(50)

W(50) = 19 µs +(50\50)*18.5 µs+(50/10)*5 ms+ (50/20)* 21.6 µs ms = 25.0915 ms

W(50) < D = 25.0915 ms <50ms , **Task 2 is Schedulable**

For Task 3 (Periodic Transmitter)

W(1)……W(100)

W(100) = 19.2 µs +(100/50)*19 µs+(100/50)*18.5 µs+(100/10)*5 ms+ (100/20)* 21.6 µs ms = 50.2022 ms

W(100) < D = 50.2022 ms <100ms , **Task 3 is Schedulable**

For Task 6 (Load 2 Simulation)

W(1)……W(100)

W(100) = 12 ms+ (100/100)*19.2 µs +(100/50)*19 µs+(100/50)*18.5 µs+(100/10)*5 ms+ (100/20)* 21.6 µs ms = 62.2022 ms

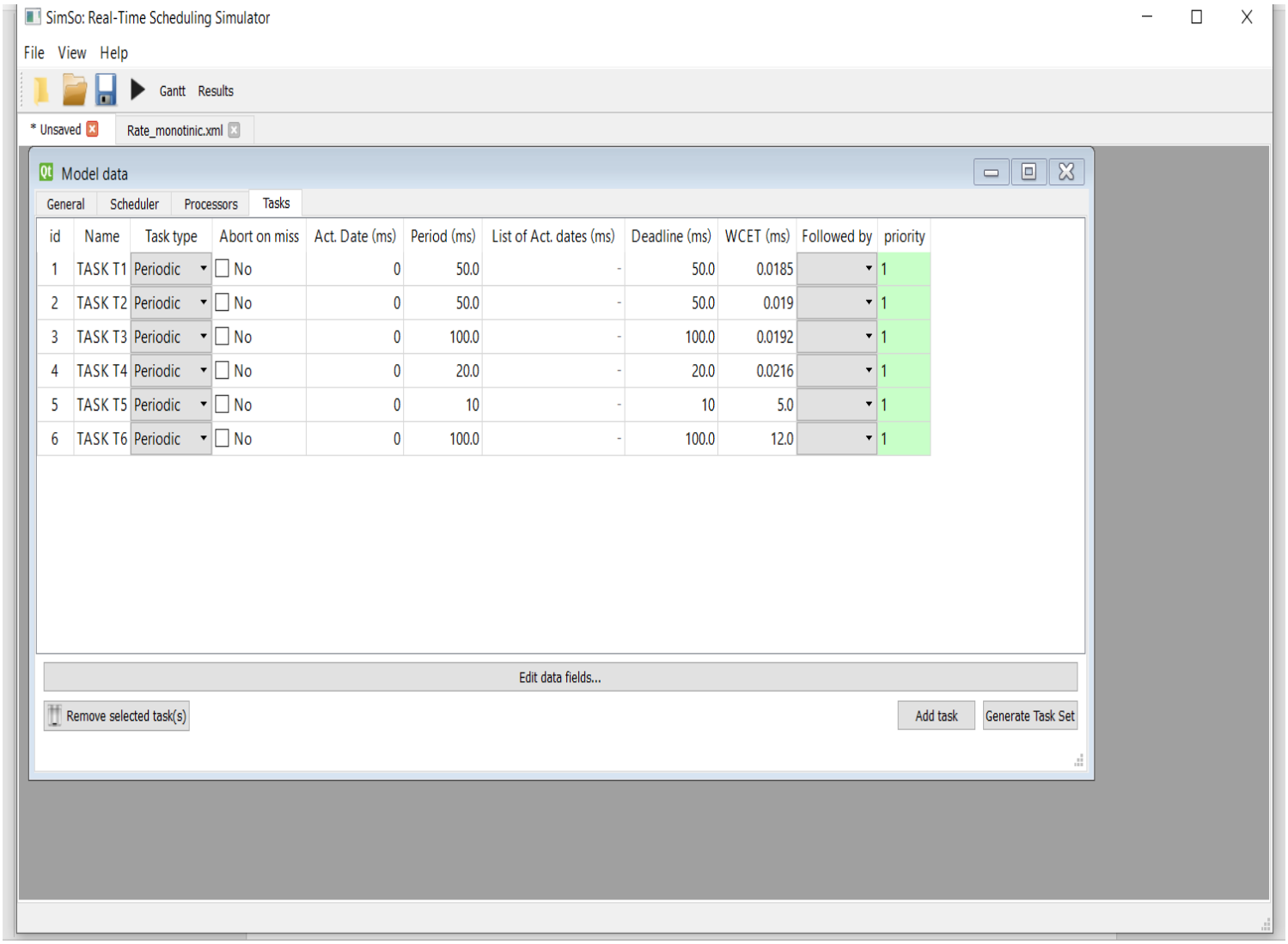
W(100) < D = 62.2022 ms <100ms , **Task 6 is Schedulable**

Method 2: SIMSO

- Using Simso offline simulator, simulate the given set of tasks assuming:

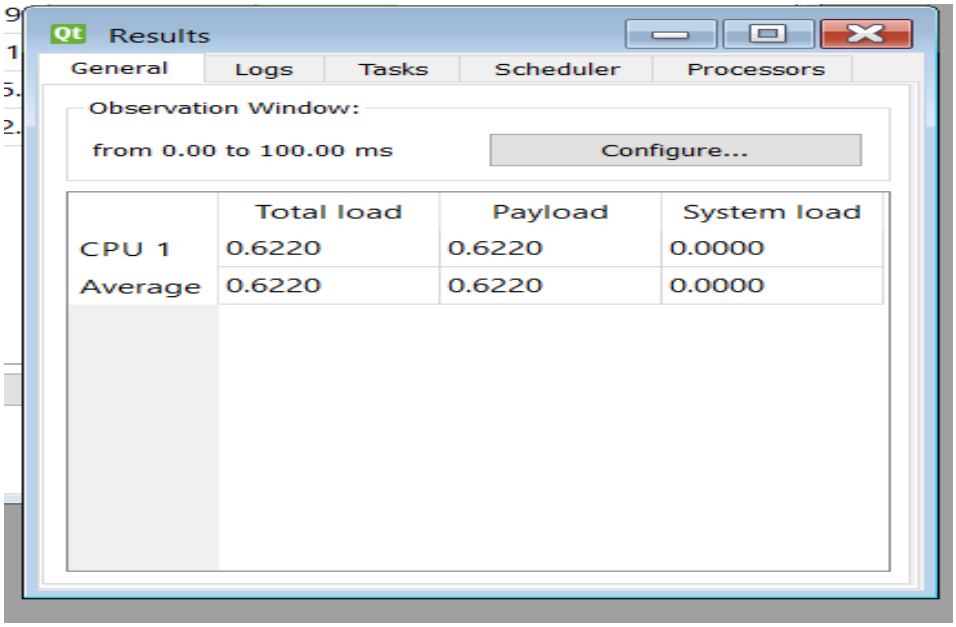
  1. Calculate Hyper-period
  2. CPU Load
  3. Schedulability Analysis

- Tasks Creation

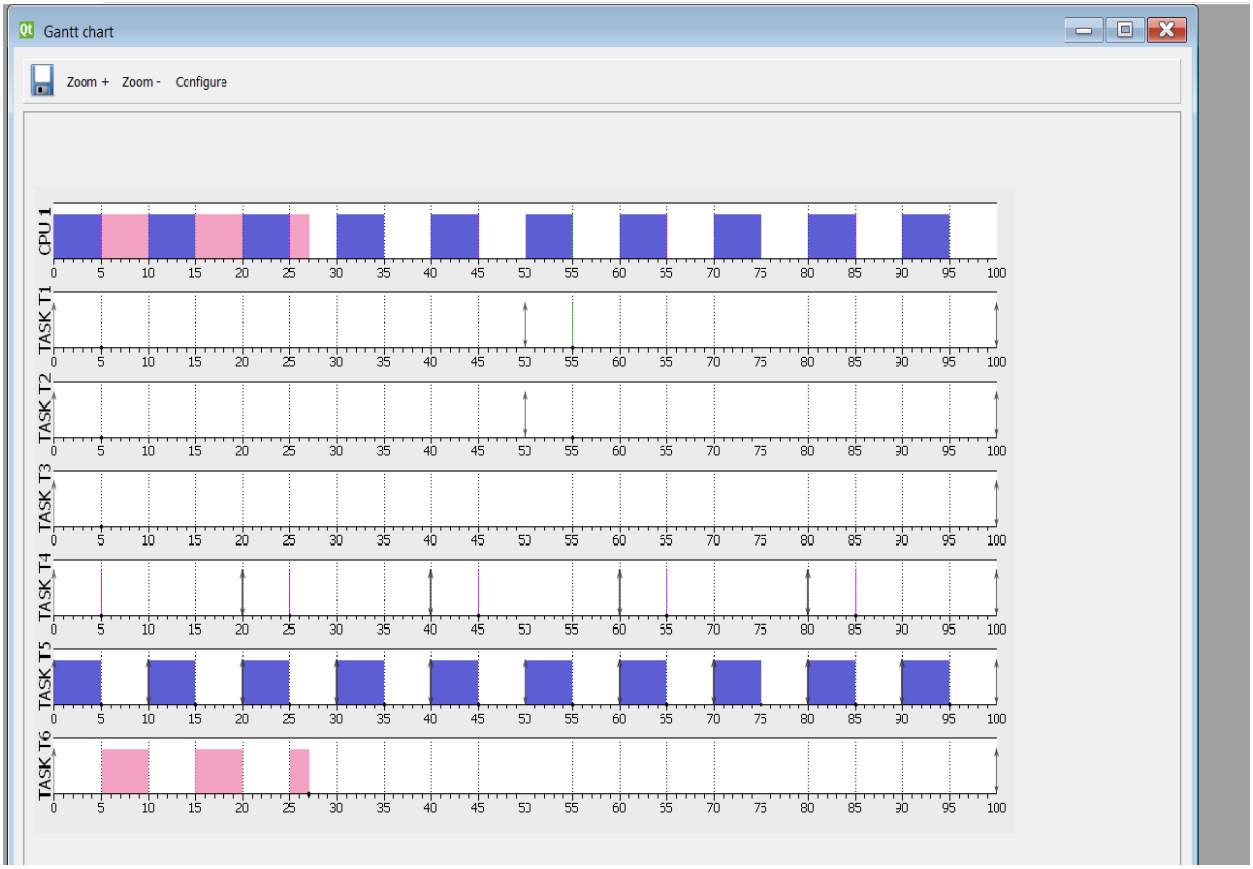


1. Calculate Hyper-period=100ms

2. CPU Load (Simso)



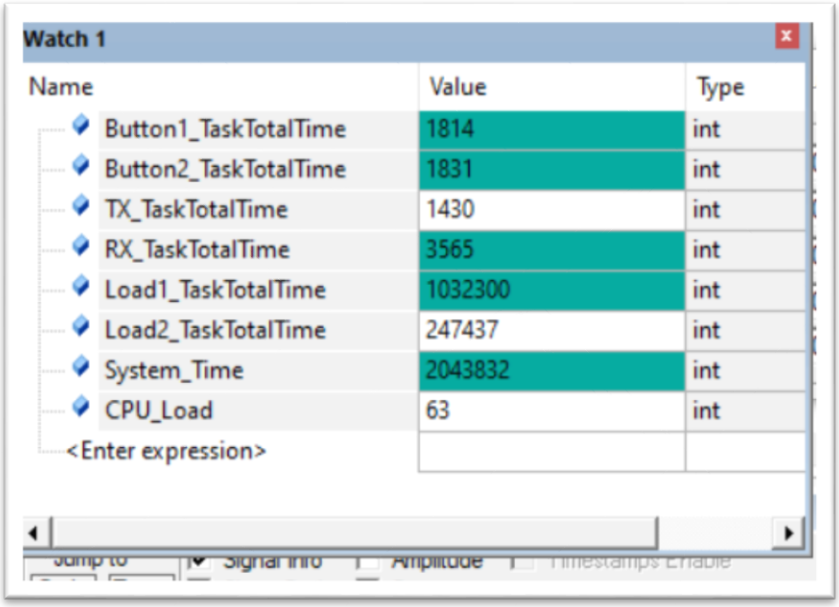
3. 3.Schedulability Analysis: system is Schedulability from Gantt Chart



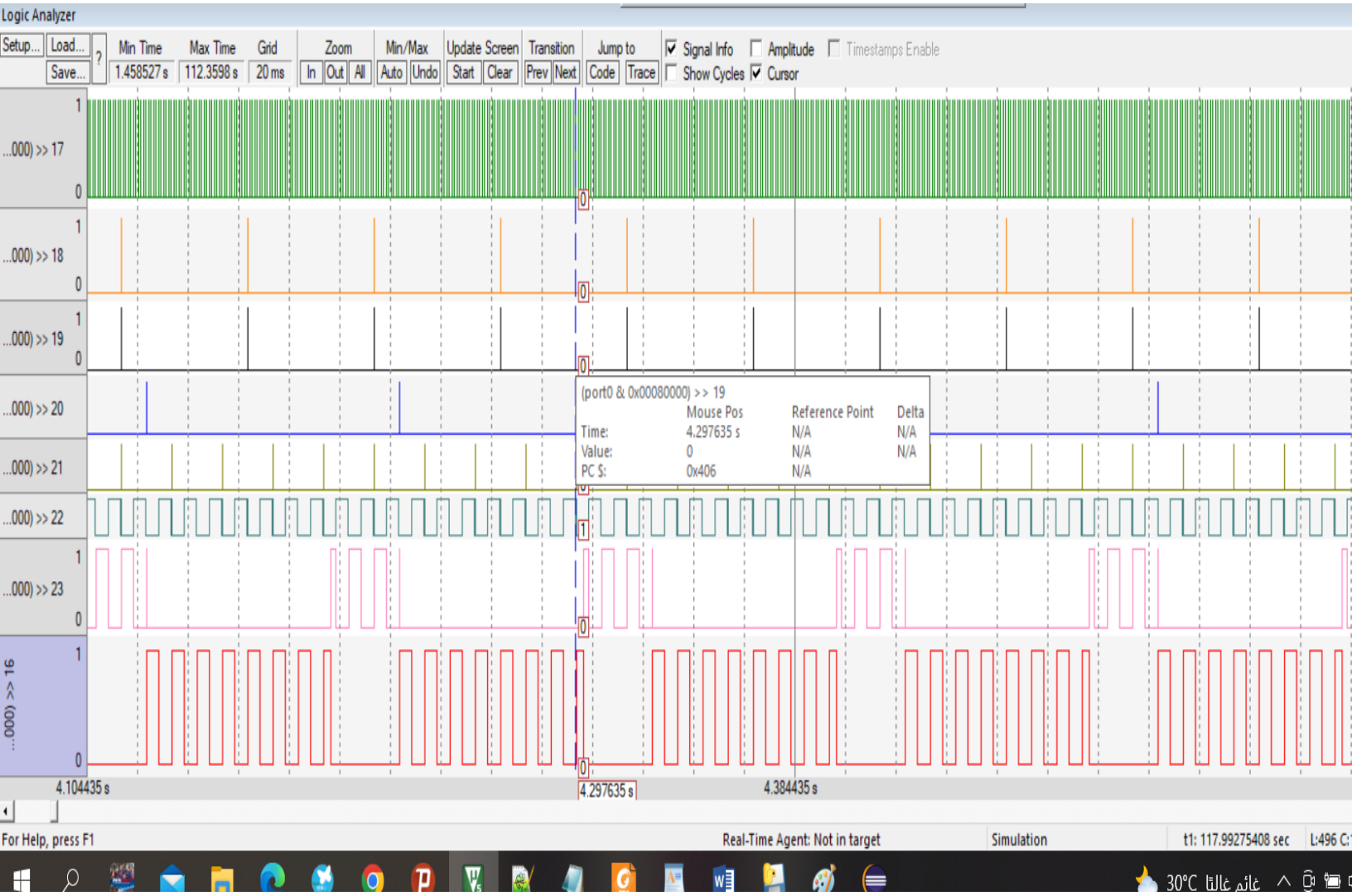
Method 3: Using Keil

Using Keil simulator in run-time and the given set of tasks

1- Calculate the CPU usage time using timer 1 and trace macros



2- Using trace macros and GPIOs, plot the execution of all tasks, tick, and the idle task on the logic analyzer"



Comment on the Results:

As We see the results of the three methods give the same **CPU load = 62.7%,** which means a successful implementation.