**Analytical Methods**

This document includes analytical methods of verifying EDF Scheduler based on FreeRTOS.

1. **System Hyperperiod**
   * This project contains 6 tasks as follows :

|  |  |  |
| --- | --- | --- |
| Task | Period | Execution Time |
| Button\_1\_Monitor | 50ms | 13us |
| Button\_2\_Monitor | 50ms | 13.2us |
| Periodic\_Transmitter | 100ms | 17.5us |
| Uart\_Receiver | 20ms | 27.35us |
| Load\_1\_Simulation | 10ms | 5ms |
| Load\_2\_Simulation | 100ms | 12ms |

* Using the above table, we can easily calculate the Hyperperiod for these tasks (the period after which all tasks repeat execution again)

Hyperperiod = 100ms

1. **CPU Load**
   * To calculate the CPU load, we need to calculate the execution time for each task multiplied by number of times these tasks came through one hyperperiod, then by summing these times and divide by the Hyperperiod we get the CPU load.
   * Using the same above table, we get :
     1. CPU load =

CPU Load = 62.2%

1. **System Schedulability**
   * This property can be determined using two methods:
     1. Rate-Monotonic utilization bound (only for RM Schedulers)
     2. Time demand analysis
   * Let’s start with Rate-Monotonic method

Rate Monotonic Utilization Bound

Diagram

Description automatically generated

* + There are two side of the above equation
    1. The right-hand side calculates the summation of ratio between the execution time of a task and the periodicity of that task
    2. The left-hand side is called URM which is considered the metric of the system schedulability
  + By comparing these two sides of equation :
    1. if the right-hand side is less than or equal to the URM term, system is schedulable
    2. Otherwise, system is not schedulable
  + By Applying this equation on our system :
    - The right-hand side :

* + - The left-hand side (URM) :
    - So, System is Schedulable
  + For now, let’s illustrate the second method

Time Demand Analysis

* + Text, letter

    Description automatically generatedThis method measures the time required against the time provided for each task
  + Let’s reorder our tasks based on priorities to match Rate-Monotonic scheduler rules
    - tasks with higher periodicity (comes faster) take higher priorities

|  |  |  |  |
| --- | --- | --- | --- |
| Priority | Task | Period | Execution Time |
| 0 | Load\_1\_Simulation | 10ms | 5ms |
| 1 | Uart\_Receiver | 20ms | 27.35us |
| 2 | Button\_1\_Monitor | 50ms | 13us |
| 2 | Button\_2\_Monitor | 50ms | 13.2us |
| 3 | Periodic\_Transmitter | 100ms | 17.5us |
| 3 | Load\_2\_Simulation | 100ms | 12ms |

\*\* Priority : lower value means higher priority

* Using the above equation, let’s calculate response time for each task taking into consideration the effect of other tasks if they have higher priority
  + - Load\_1\_Simulation :

W(10) = 5 + 0 = 5ms 5ms 🡪 Schedulable

* + - Uart\_Receiver :   
      W(20) = + 5\*() = 10.027ms 20ms 🡪 Schedulable
    - Button\_1\_Monitor :

W(50) =+ + 5\*() = 25.095ms 50ms

🡪 Schedulable

* + - Button\_2\_Monitor :

W(50) =+ + 5\*() = 25.108ms 50ms

🡪 Schedulable

* + - Periodic\_Transmitter :

W(100) = ++ + (= 50.188ms 100ms

🡪 Schedulable

* + - Load\_2\_Simulation :

W(100) = 12 + ++ + (= 62.206ms 100ms

🡪 Schedulable

* + Using the above results, all tasks are schedulable

System is Schedulable

SIMSO Offline Results

In this part we will show simso offline simulator results for our project including same tasks discussed above in the previous part. This step mainly purpose is to verify the above results of analytical approach and see if they match or not.

Graphical user interface

Description automatically generated with low confidence1.

* + - As we see that CPU load = 62.2% which similar to analytical approach

A picture containing timeline

Description automatically generated2.

* + Here we can see that first task (Load\_1\_Simulation) executes for 5ms and happen every 10ms with highest priority, which is totally expected
  + Similarly with rest tasks we can see each task and when to execute
  + Task2, Task3, Task4 and Task5 have very small execution time so it’s not clear enough on simulation

Background pattern

Description automatically generated

* + This is a clearer screenshot which clarifies those short execution time tasks.

Table

Description automatically generated

* + Here is the whole situation including all tasks (short time tasks will not appear due to zoom out)
  + We can see the case of executing task6 while task1 came (higher priority and earlier deadline) , so task1 preempted task6 and this happens three times every execution of task6
  + From above screenshots using simso, and the above results of analytical approach, we found that both approaches lead to same results and there is a matching between them.

Keil Simulator Results

In this part we will show Keil simulator results for our project including same all tasks discussed above in the two previous parts. Using timer1 and macro tracing we will calculate the CPU usage (load) and verify that all tasks run well with correct time constraints.

Timeline

Description automatically generated1.

IDLE Task

Load\_2\_Simulation

Load\_1\_Simulation

Uart\_Receiver

Periodic\_Transmitter

Button\_2\_Monitor

Button\_1\_Monitor

Tick

Here we can see that CPU\_Load is about 62.5% (saturate between 62% and 63%).

This result matches our previous two approaches (analytical and simso simulator)

Line chart

Description automatically generated with medium confidence2.

This shows the execution time

of Button\_1\_Monitor task in run time.

Which is around 13us.

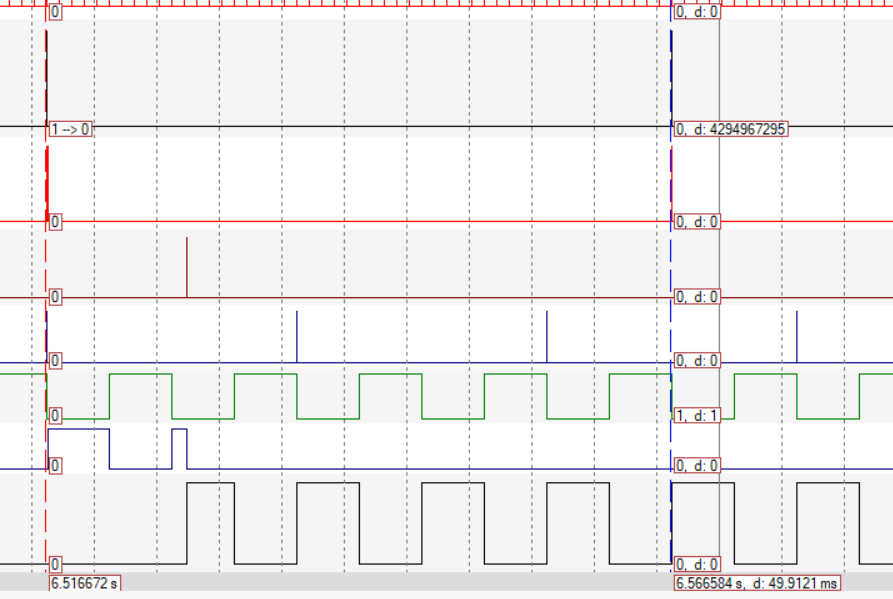
Graphical user interface, chart

Description automatically generated3.

This shows the execution time

of Button\_2\_Monitor task in run time.

Which is around 13us.

4.

This shows That both

Button\_1\_Monitor and

Button\_2\_Monitor tasks

periodicity is about 50ms

Graphical user interface, application, table

Description automatically generated5.

This shows the execution time of Periodic\_Transmitter task in run time.

Which is around 17.5us.

Chart

Description automatically generated6.

This shows the periodicity of Periodic\_Transmitter task

Which is around 100ms

Calendar

Description automatically generated7.

This shows the execution time of Uart\_Receiver task in run time.

Which is around 27.5us.

A picture containing graphical user interface

Description automatically generated8.

This shows the periodicity of Uart\_Receiver task

Which is around 20ms

A picture containing calendar

Description automatically generated9.

This shows the execution time of Load\_1\_Simulation task in run time.

Which is around 5ms as required.

A picture containing graphical user interface

Description automatically generated10.

This shows the periodicity of

Load\_1\_Simulation task

Which is around 10ms

11.

Graphical user interface

Description automatically generated with medium confidence

This case shows that Load\_2\_Simulation task (the blue one) is preempted by the above task (Load\_2\_Simulation) three times, so total execution time for Load\_2\_Simulation task = 27 – (3\*5) = 12ms which equals the time required in specifications.

12.

Calendar

Description automatically generated with medium confidence

This one shows the periodicity of Load\_2\_Simulation task (the blue one)  
which is 100ms.