CSE-438 – Embedded Systems Programming
Project 4: Real Time Analysis of the application

- **Identifying the tasks and measuring their execution time:**
  The application requires three periodic tasks to execute correctly. All the three periodic tasks have a time period of 10 milliseconds, which is the refresh rate of the led board. Two of the tasks execute in the user space, and one in the kernel space. The tasks are as follows:

  - **Task 1 (Executed in user context):**
    This task is used to read the input events and update the current angle accordingly. It uses the complementary filter to calculate the current angle based on the gyroscope and accelerometer readings. We measure its total execution time and the number of iterations, and divide the two to find the average execution time. The calculated time is as follows:

    Total Execution time: 16334
    Number of Iterations: 5140
    Average Execution time = 16334/5140 = 3.177 µS

  - **Task 2 (Executed in user context):**
    This task is used to calculate the current coordinates of the ball (Lighted LED), and update the LED grid accordingly. SPI interface is used to update the LED grid. This task has a number of sleeps in it. So, to measure the execution time, we decrease the total execution time by the time it sleeps every time. This gives us an accurate execution time, and gives us a good idea of the schedulability of the tasks in the real time analysis. The calculated times are as follows:

    Total Execution time: 8068558
    Number of Iterations: 3922
    Average time = 8068558/3922 = 2057.052 µS

  - **Task 3 (Executed in the kernel context)**
    This task is used to read the gyroscope and accelerometer readings and post input events. I2C communication protocol is used to read the registers of MPU6050. The task has three sleeps, so we measure its execution time by separately measuring the execution times of four segments of the task. The times calculated are as follows:

    Total Execution time (Segment 1): 1506896 µS
    Total Execution time (Segment 2): 2515154 µS
    Total Execution time (Segment 3): 1500478 µS
    Total Execution time (Segment 4): 2601015 µS
    Number of iterations: 2447

    Average execution time of the tasks: 8123543/2447 = 3319.797 µS

- **Real Time Analysis**
  For the purpose of real time analysis, we use a pessimistic approximation for the worst case execution time. This is done to make sure that the execution time of the task never exceeds the worst case execution. The approximated worst case execution time of the tasks are as follows:

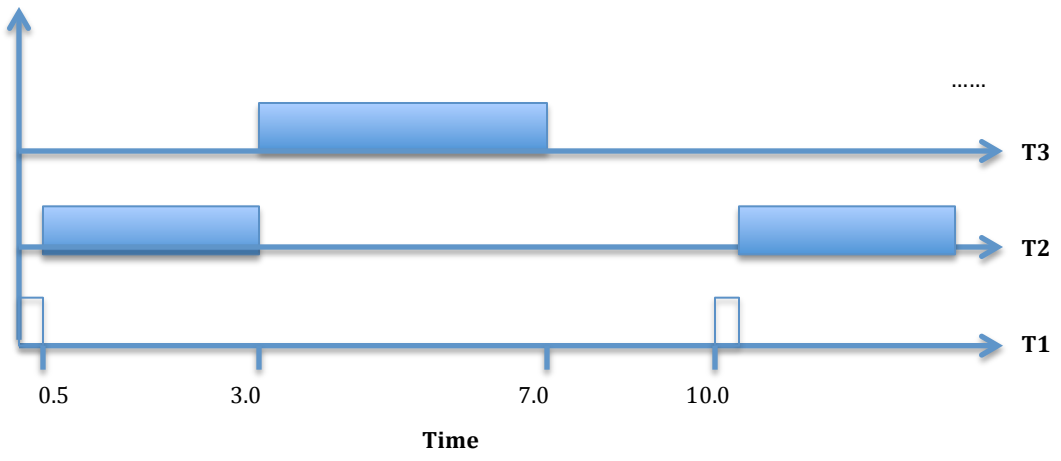
  T1: 10 µS
  T2: 2500 µS
  T3: 4000 µS

  The period of all the three tasks is 10000 µS. So we can calculate the total CPU utilization (U) of the application as:
  U = 10/10000 + 2500/10000 + 4000/10000 = 0.6510

  To check whether it can be scheduled using Rate Monotonic (fixed Priority) scheduling algorithm, we do a utilization bound test. For the three tasks to be schedulable,
  $U <= n (2^{1/n} -1) = 3 (2^{1/3} -1) = 0.771$ (which is true)

  Therefore, we can safely say that the three tasks are schedulable using Rate Monotonic Scheduling Algorithm.

- **Designing a schedule:**
  We can design a simple clock based static schedule for the three tasks. We keep the minor cycle to be 0.5 milliseconds, and the major cycle to be 10 milliseconds. We keep the deadlines of all the tasks to be 10 milliseconds. We need to all the tasks in one major cycle to make sure that all the tasks finish execution before their deadlines. A valid clock based static schedule for the three tasks can be seen in the figure below:



- **Conditions under which deadlines could be missed:**
  The task can miss deadlines if the total utilization is more than the maximum schedulable utilization. If the total utilization of the tasks increases 0.771, we cannot be certain if all the tasks can meet the deadline. We will need to do a response time analysis to find if the

tasks will meet the deadlines. If the total utilization increases more than 1, we can be certain that the tasks will not meet the deadlines.
The utilization can increase if any of the tasks take more than their worst case execution time to finish its execution. It is possible that task 2 and task 3 take more time that their worst case execution time, due to communication errors or loose circuit connections. It is unlikely that task 1 will miss its deadline, as it does not involve communications with an external chip.