CM0605 - Embedded Systems Engineering

Liam Brand

Chapter 1

Real Time Scheduling

1.0.1 Basic Scheduling Analysis

(a)

The utilisation of a task set is given by

$$U = \sum_{i=1}^{N} \frac{C_\mathrm{i}}{T_\mathrm{i}}$$

Where N is the number of tasks, C is the task's worst-case execution time and T is the task's period. The given task set's utilization is approximately 0.99.

(b)

First a utilisation-based schedulability test can be used to determine whether or not the task set is schedulable using rate-monotonic priority assignment.

The necessary test is the following. If this test is negative, the task set is definitely not schedulable.

$$U \leq 1$$

The test is positive, so the sufficient schedulability test can be carried about. The sufficient test is the following, where n is the number of tasks.

$$U \leq n(2^{\frac{1}{n}} - 1)$$

For the given task set with 6 tasks, this formula equals approximately 0.73, meaning this test is negative. If the necessary schedulability test is positive but the sufficient schedulability test is negative, a better schedulability test is needed.

Question c

Question d

Question e

1.0.2 Scheduling with Shared Resources

Question a

Question b

Chapter 2

A Distributed Real-time System

Chapter 3

Reliability

3.1 Section 1

3.1.1 Question a

Fail-Operational

Fail-operational is when a system is still capable of full performance in the presence of faults, with no external signs of the fault manifesting.

Fail-Active

Fail-active systems can continue their operation when a fault is encountered, but will do so at a reduced performance. This is the least applicable system to the production line, as a reduced performance will result in a lower production quality and the possibiliy of the system's mission goals not being met, but also the possibility of personnel being injured.

Fail-Safe

Fail-safe systems will cease operation upon encountering a fault, and enter a safe mode. For the car production line, this will completely minimize risk to human life but production will suffer the most as each encountered fault will result in the system entering its safe mode. Restarting the system and bringing it back to full operational capacity could take a while, and this will result in a potential failure of the system's mission objectives.

High Availability

High availability systems cease operation upon encountering a fault, but must be returned to operation as quick as possible, this can involve hot swapping system units whilst the system is operating. These systems allow for failures, and aim to achieve a high mean time of operation rather than a long continuous

time of operation. The goal here is not to avoid faults, but to minimize time spent rectifying them so that the overall system operation time is as high as possible[1]. The availability of these system can be measured by the following, where MTBF is mean time between failures and MTTR is mean time to repair.

$$Availability = MTBF/(MTBF + MTTR)$$

Of the different types of fault-tolerance, high-availability is the most appropriate for the car production line. Whilst the operation of the production line is regrettable and will likely result in financial losses and other goals being missed such as a target for the amount of cars manufactured, the potential for injury to personnel is by far the most severe consequence of system failure. The potential for harm is why continued system operation shouldn't be implemented in the event of a fault, but high-availability will place emphasis on getting the system back up and running so cars can be properly created again. This will help minimize the impact on the system's mission by reducing its downtime.

3.1.2 Question b

Potential Risks

The nature of the car assembly line means any single failure will cause the entire system to fail, so in terms of the network only a single connected node in the distributed system needs to experience a problem for the entire system to stop working.

Timing will also be essential within the assembly line to ensure the robotic tools perform their jobs properly, so there is a risk of performance overhead negatively impacting the system too. Networked processors that are in close proximity will communicate incredibly quickly, but processors that are further away from the network will take longer to communicate and this will need to be accounted for otherwise tools may not behave as expected and, should any issues occur, the entire system will fail as previously discussed. In a similar vain, timing will need to be considered to ensure that all operations across the network are in synchronisation with each other. Messages arriving on time is a good start, but this doesn't guarantee that all of the production line robotics are cooperating as expected.

Architecture Recommendation

3.1.3 Question c

Bibliography

[1] Jim Gray and Daniel P. Siewiorek. High-availability computer systems. *Computer*, 24(9):39–48, 1991.