Rochester Institute of Technology
Real Time and Embedded Systems
Project 4

Design and implement an embedded, stand-alone QNX Neutrino program to simulate the workflow in a typical banking environment -- single queue with queuing to a multi-threaded server or three separate server processes.

Problem Statement:

- Customers enter the bank to transact business on a regular basis. Each new customer arrives every one to four minutes, based on a uniform random distribution. Each new customer enters a single queue of all customers.
- Three tellers are available to service customers in the queue. As tellers become available, customers leave the queue, approach the teller and conduct their business. Each customer requires between 30 seconds and 8 minutes for their transaction with the teller. The time required for each transaction is based on a uniform random distribution.
- The bank is open for business between the hours of 9:00am and 4:00pm. Customers begin entering when the bank opens in the morning, and stop entering when the bank closes in the afternoon. Customers in the queue at closing time remain in the queue until tellers are available to complete their transactions.

Metrics:

To monitor the performance of the business, metrics are gathered and reported at the end of the day. The metrics are:

1. The total number of customers serviced during the day.
2. The average time each customer spends waiting in the queue.
3. The average time each customer spends with the teller.
4. The average time tellers wait for customers.
5. The maximum customer wait time in the queue.
6. The maximum wait time for tellers waiting for customers.
7. The maximum transaction time for the tellers.
8. The maximum depth of the customer queue.
9. Grad student additional metrics:
    - Number of breaks for each of the three tellers
    - Average break time for each of the three tellers
    - Longest break time for each of the three tellers
    - Shortest break time for each of the three tellers

Design Constraints:

- You are free to use any QNX supported concurrency mechanism to implement this lab. Be sure to describe your selected architecture in your lab report.
- You may want to create a semaphore to communicate the number of available tellers but other options include QNX message sending and reply blocking.
- Each teller is modeled as an independent thread in the Teller Server or as three separate processes.
- The simulation parameters as described in the Problem Statement can be “hard-coded” as internal constants. However, only define those values in one location – a header file is strongly recommended.
- The simulation time is scaled such that 100 milliseconds of absolute clock time represents 1 minute of simulation clock. Therefore, this program will run about 7 hours x 60 minutes /hour * 0.1 seconds / minute.
- The output must be presented in simulation clock time (9 AM through closing time around 4 PM).
- In order to get the timing right I strongly recommend using timers and then waiting for the timer to complete. Using CPU consuming library functions like usleep are discouraged. Substantially inaccurate simulations will be penalized.

Graduate Students:

- Include random breaks for each of the tellers.
- Each teller will take a break every 30 to 60 minutes for 1 to 4 minutes. If a teller break time occurs while serving a customer they will go on break as soon as they finish the current customer.
- The next break for a teller occurs from 30 to 60 minutes from when they started their previous break.
- A break can only occur after the completion of the current customer transaction.
- The break timing and duration is based on a random uniform distribution.

Report:

In addition to the demonstration of your project, a brief report is required to describe your design. Refer to the Report Specifications for the required content and format.

Be sure to include one run of your output in your report. Please submit your report as either a Word compatible document or a PDF document in the project dropbox. Do not put it inside a zip file or other archive.

Your source code (the .c files you created along with any .h files) must be included in your electronic submission. Files generated by the compiler do not need to be submitted.

Due Dates:

Refer to the class schedule for the due dates.
Part 1 is a software design document. This must include appropriate diagrams and explanation to cover the following topics:

- Design of the customer queue including randomized customer arrival time and randomized amount of time required at the teller window.
- Description of how you measure the customer metrics
- How you assign a customer to a teller (what is your algorithm?)
- How you measure all teller metrics
- For grad students, describe the design of the break times including how you collect required additional metrics.

Grading Criteria:

- Program Operation and Demo – 40%
    - Hardware setup is orderly and well organized – 10%
    - Demo sheet functions all completed – 30%
    - Demo operates without faults or restarts – 10% (except for POST verification)
- Program Design --- 25%
    - A detailed software design is required as a first deliverable. This is 15 of the 25 points for program design. This needs to be also included in the final report.
    - Proper initialization
    - Correct use of functions (no copy/paste/edit slightly)
    - Separation of hardware related code from pure software (e.g. the results reporting code)
- Source Code Structure and Readability – 10%
    - Appropriate use of white space – 2%
    - Consistent and good indentation – 2%
    - Appropriate comments at the function and paragraph levels (such as a for loop) – 2%
    - Following C style guide (good names, etc.)
- Report Content – 25%

- Report is at least 2 pages (not counting pictures, cover page, diagrams) – 5%
- Demonstrates team understands the problem, solution, and technology (hardware and software) – 10%
- Report contains all required sections per the report guidelines – 10%