EE525 Course Introduction

2020 Fall

The course focuses on methods of design and validation of real-time systems. "A real time system is one in which the correctness of the system depends not only on the logical result of computation, but also on the time at which the results are generated.[1]" The main problem that real-time systems course deals with is: "How to design and validate a system such that a bounded response time to external events can be guaranteed." This problem is divided into categories of task scheduling on single processors, task allocation on multiple processors and inter processor communication, all under the guarantee of latency constraints.

Today, real-time systems theory is one of the main building blocks of the concept of **Internet of Things (IoT)** where embedded systems interacting with the world cooperate over communication networks to perform tasks. Almost all embedded systems have some real-time aspect since they interact with physical plants. Some examples of real-time systems are telecom, multimedia and transaction applications, transportation and utility infrastructure, medical, military and aerospace applications, as well as consumer goods, such as multimedia and communication devices, vehicle electronics, etc.

Syllabus:

- Definition and requirements of real-time systems
- Resource allocation methods.
- Real-time scheduling:
  - Fundamental scheduling algorithms, proofs of optimality.
  - Scheduling of periodic tasks under ideal conditions and real world constraints .
  - Scheduling of non periodic tasks with soft timing constraints and strict timing constraints.
  - Multiprocessor task allocation and scheduling
- Real-time communication and other relevant subjects as requested by a significant number of enrolled students may be included.

[1] J. Stankovic, "Misconceptions of Real-Time Computing", IEEE Computer, Vol 21, No 10, pp 10–19, 1988

Detailed Syllabus:

- **Week 1** Introduction to real-time systems
- **Week 2** Description of terminology and elementary concepts
- **Week 3** Common approaches to scheduling. Cyclic schedules
- **Week 4** Rate monotonic scheduling (RM). Critical instant, busy interval, worst case response time and associated topics.
- **Week 5** Schedulability tests for rate monotonic algorithm.
- **Week 6** Schedulability tests under transient overloads, and other issues encountered in real applications.
- **Week 7** Earliest deadline first scheduling (EDF). Demonstration of optimality of EDF. Other dynamic priority scheduling algorithms.
- **Week 8** Scheduling of non periodic tasks. Introduction and general concepts. Task server methods. Deferrable servers.
- **Week 9** Servers for aperiodic task scheduling (cont.)
- **Week 10** Slack stealing methods.
- **Week 11** Dynamic priority scheduling of aperiodic tasks. Constant utilization server. Fairness and starvation concepts.
- **Week 12** Sporadic task scheduling methods.
- **Week 13** Multiprocessor task allocation and scheduling introduction. Task assignment to minimize resource requirements such as communication.
- **Week 14** Multiprocessor task allocation and scheduling (cont.). End to end scheduling, ISP and greedy algorithms. Presentation of term projects.

Project:

Due to the remote nature of this year's course, the project needs also to be done on-line. We will be simulating an embedded computer within Matlab, using the TrueTime toolbox. (See Sec: "Software" below.)

For the project, a distributed data acquisition system will be built. It consists of 3 coputer nodes and a computer network, as seen in Fig. 1. The computer Node 1 samples two periodic data streams of different frequencies, and should run periodic tasks. On the other hand, node Node 2 captures events represented by pulses generated at random instants, and therefore must run aperiodic tasks. Finally, Node 3 is responsible for the collection and display of data. It should run a task which executes every time a data sample is collected from any of Nodes 1,2 and plot the received data to the "Data display".

You are expected to:

- Allocate sufficient executuion times for the tasks running at each node. Show that with the periods compatible with the Nyquist sampling theorem, the tasks are schedulable on each of the processors.
- Show the task schedule on each processor on a graph.
- Select and implement a suitable communication network and explain your reasoning.
- Show the data packet transmissions on the network, and determine the network parameters especially bandwith, such that the data is not delayed.

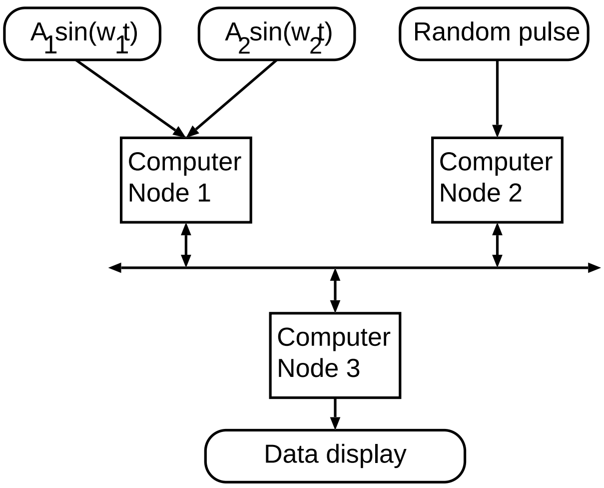

Figure 1: The system to be modeled in TrueTime toolbox.

Exam schedule

There will be 7 short exams during the semester. Each exam will be held at the end of Thursday's lecture (except 2nd exam) during the last 15 minutes, and will cover everything from the start of the class until the end of previous lecture. The exam list is:

- Oct 15^{th}: Short exam
- Oct 26^{th}: Short exam (This one is on a Monday!)
- Nov 12^{th}: Short exam
- Nov 26^{th}: Short exam
- Dec 10^{th}: Short exam
- Dec 24^{th}: Short exam
- Jan 7^{th}: Short exam

Books:

We will make use of the following books. Mainly Krishna and Shin for overview, and Liu for detail work.

- Jane W. S. Liu "Real-Time Systems", Prentice Hall, ISBN 0130996513, 2000. Main course book.
- C. M. Krishna, K. G. Shin, "Real-Time Systems", McGraw Hill, ISBN 0070570434, 1997. Elementary subjects; supplementary book.

Software:

You are expected to use Matlab with Simulink and TrueTime.
Installation of Matlab can be found at University internal website `mysu`.
Installation of TrueTime 2.0 can be done from:
`http://www.control.lth.se/research/tools-and-software/truetime/`
It has been checked on Matlab v2019b. It should work on other Matlab versions.

Grading:

The grading will be done over a weighted average of short exams and a project. The weights are biven below, but are subject to slight changes at the end of the semester to maximize the class average.

- Each short exam: 11 % ($\times 7 = 77\%$)
- Project: 23%