Laboratory Exercises on Task Scheduling in Real-Time-Systems course

Subramaniam Ganesan
Department of Electrical and Computer Engineering
Oakland University
Rochester, MI 48309
ganesan@oakland.edu

Abstract

One of the objectives of a graduate level course on "Real time systems" is: ´Know the various task assignment and scheduling methods. For example, Rate Monotonic (RM) and Earliest Dead line First (EDF) scheduling

A graduate level Real-time-systems course emphasizes hard and soft real time computer system design for uniprocessor embedded system applications and distributed real time systems. Topics taught every week, home work and laboratory assignments are organized to meet the weekly course objectives.

We describe here a few simulation software tool, author developed excel based task schedule timing diagram viewer and a software tool developed using Lab View for scheduling tasks with different algorithms. Laboratory assignments with microprocessor board with a simple Real time operating system development tool to teach the scheduling are described. A few lab experiments to teach real-time task scheduling concepts are explained. We emphasize active learning instead of passive learning. We describe also how the objectives of the course are assessed by students and by peer review of the complete course content.

Introduction

Real time system course is a graduate level course. It emphasizes hard and soft real time computer system design for uniprocessor embedded system applications and distributed systems [1-6]. Topics covered include characterizing real-time systems, performance measure, task assigning, scheduling, Fault tolerant scheduling, run-time error handling, run-time support, kernel, real time databases, real-time communication, software development techniques; practical applications. Objectives of the Real-Time Systems course reflect what we expect the students to learn from the course. We make sure that it is possible to assess the objectives at the end by internal and external experts. We make sure that the teaching activities and feedback from the students are synchronized with the objectives

Course Objectives are:

Upon completion of this course students will be able to:

1. Know the definition and characteristics of Real Time systems
2. Know the various task assignment and scheduling methods. For example RM and EDF scheduling.
3. Become familiar with Real Time system development tools like Matlab RT tool box, ETAS tools
4. Know the important characteristics of Real Time Operating System
5. Know about the RT System requirement, design, and performance analysis.

In the following sections, we show how we make sure that the students learn the objective number two on scheduling. Details of course topics are given in the appendix A.

Scheduling

There are many task scheduling algorithms and each of them makes sure that the tasks are schedulable and they meet the expected deadline [1, 7]. Three most popular algorithms are briefly mentioned below.

Rate Monotonic Scheduling Algorithm

The rate-monotonic scheduling algorithm schedules periodic tasks using a static priority policy with preemption. If a lower-priority process is running and a higher-priority process becomes available to run, it will preempt the lower-priority process. The shorter the deadline, the higher the priority

Deadline Monotonic Scheduling Algorithm

Deadline Monotonic Scheduling Algorithm schedules periodic tasks or aperiodic tasks using a static priority policy with preemption. For each process, the shorter the relative deadline, the higher the priority.

Earliest-Deadline-First Scheduling algorithm

Earliest-Deadline-First scheduling dynamically assigns priorities according to deadline. The earliest the deadline, the higher the priority. Algorithm: At each moment of time t, schedule the task whose deadline is closest to t. Ties are resolved arbitrarily

Scheduling Simulators

We describe below task scheduling simulators that are free and are useful for teaching scheduling. One is called CPUSS [9] and the other is Cheddar [10].

CPU Scheduling Simulator (CPUSS) allows you to quickly and easily design and gather metrics for custom CPU scheduling strategies. CPUSS records a number of metrics about the selected scheduling algorithm, like execution time, mean response time, idle CPU time etc.

CPUSS support the following scheduling strategies:

- First Come First Served
- Round Robin (time quantum can be defined)
- Shortest Job First
- Priority First
- SJF with Priority Elevation rule (threshold can be defined)

CPUSSRG allows to quickly and efficiently generate a report and view some of the key statistics from the simulation (average, standard deviation, waiting times, CPU utilization and others). One can run the simulation many times for further analysis.

Cheddar Real Time Simulator [10].

Cheddar is a free real-time scheduling tool. Cheddar allows you to model software architectures of real-time systems and to check the ability to schedule or others performance criteria. With the schedulability analysis tool, schedulability of tasks can be tested. New real-time scheduling policies or task models can be analyzed.
Cheddar is composed of two independent parts: an editor used to model the real-time system to analyze, and a framework to perform analysis. The editor allows to describe systems composed of several cores, processors which own tasks, shared resources, buffers and which may exchange messages or communication with buffers. From scheduling simulation one can get the following information.

1. Worst/best/average task response times, task missed deadlines
2. Number of preemption, number of context switch
3. Worst/best/average shared resource blocking time
4. Deadlock and priority inversion
5. Worst/average buffer utilization factor, message worst/average waiting time
6. Apply feasibility tests on tasks, buffers and shares resources :
7. Compute worst case task response time on periodic task set
8. Memory footprint of software entities
9. Task and resource priority assignment :
10. Classical Rate Monotonic, Deadline Monotonic, task priority assignment
11. Task priority assignment according to CRPD
12. Shared resource ceiling priority assignment (for PCP like policies)
13. Features to allow users to define and handle their own policies :
14. User-defined scheduling policies (based on pipeline models or automaton models)
15. Partitioning algorithms for periodic task set :
16. Best fit policy
17. General Task fit policy
18. Fist fit policy

19. Small fit policy
20. Next fit policy

Use of Other Software Tools

The students use the following tools to design, model and simulate for testing. Real-Time Workshop® is an extension of capabilities of Simulink® and MATLAB® that automatically generates, packages and compiles source code from Simulink models to create real-time software applications on a variety of system with code generator. The ETAS ASCET product family provides all the software tools needed to successfully develop model-based application software and generate C code [11]. Through a combination of static analysis and testing, designs can be validated efficiently early in the development lifecycle. The ETAS INTECRIO products offer comprehensive, reliable prototyping of automotive electronic functions. The rapid prototyping add-on (INTECRIO-RP) enables real-time function validation on the ETAS prototyping hardware devices.

Task Scheduling Lab exercises

There are many low cost microprocessor boards in the market. We chose a simple board from STMicroelectronics, that has many STM32 family of 32-bit Flash microcontrollers and application development boards [12]. Selected STM32 Nucleo-144 boards include Ethernet, as well as USB FS OTG ports to ease connections to local/wide area networks. The STM32 Nucleo-144 boards allow us to rapidly test, optimize, and develop new applications. Microcontroller development kit, MDK lite is a free, but code size limited and is good for student's use. Figure 1 shows the MDK development kit down load web page with more details. The boards use ARM, Cortex-M, or Cortex-R based microcontroller devices [13, 14]. MDK includes the µVision IDE/Debugger, ARM C/C++ Compiler, and essential middleware components. It's easy to learn and use. Keil RTX is a deterministic, small footprint real-time operating system (with source code), RTOS. It supports a few task scheduler, Execution Profiler and Performance Analyzer. Numerous example projects on the web help one to quickly become familiar with MDK-ARM's powerful, built-in features. Figure 2 shows STM 32 F746 with Cortex MT processor board and touch screen display used by students in this course (costs less than $50).

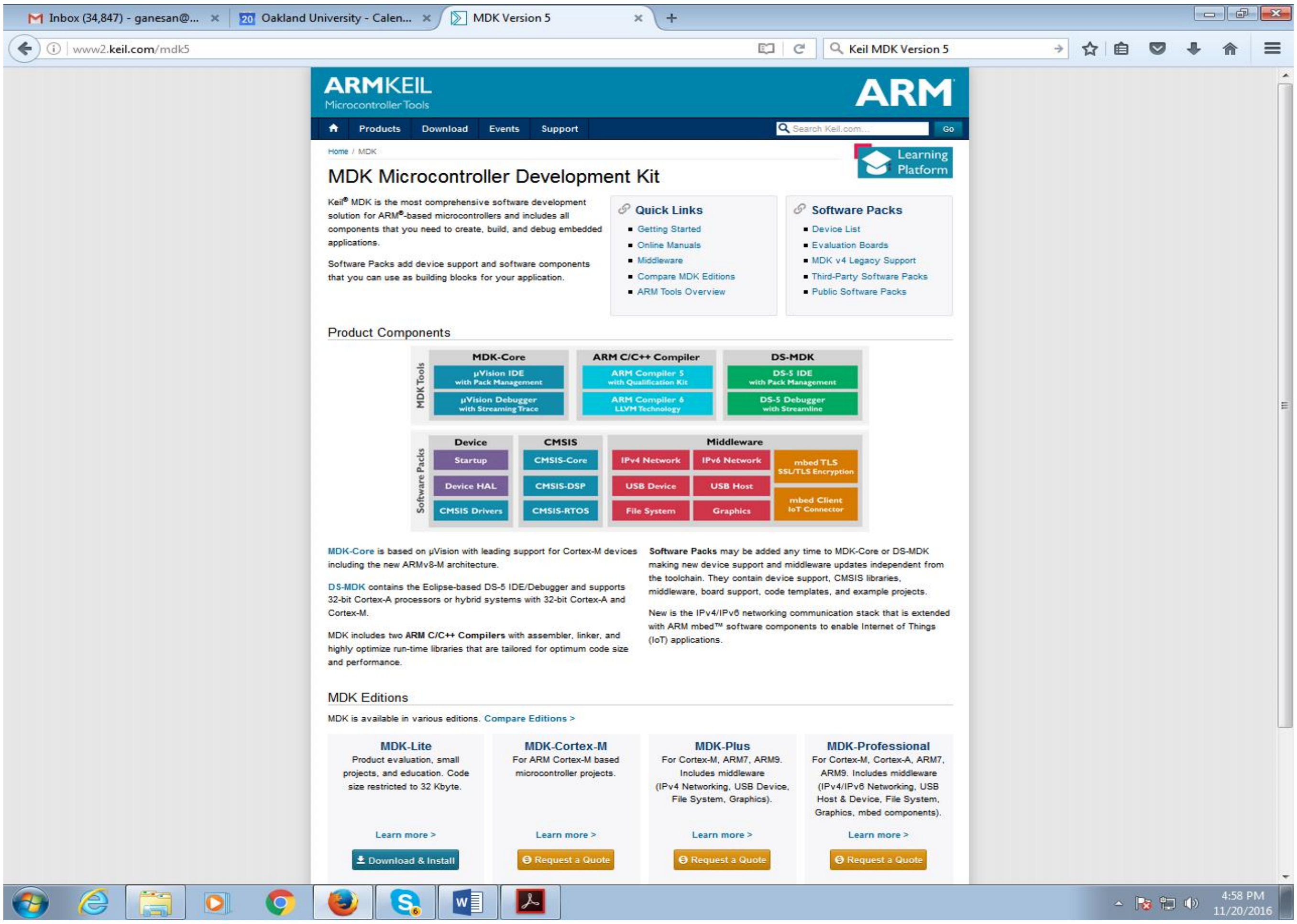

Figure 1. MDK Microcontroller Development Kit details.

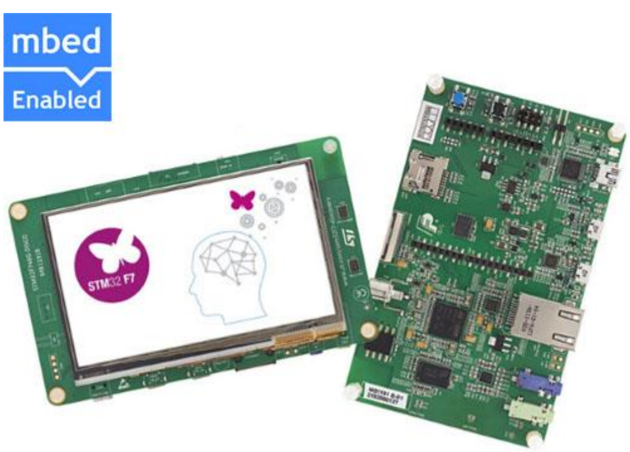

Figure 2. 32F746GDISCOVERY board with LCD touch screen display

Task Scheduling in Automotive

Today's vehicles have nearly 50 or more Electronic Control Units (ECUs), to control various functions (Example, Engine controller, Transmission controller, Display controller, Anti-lock brakes, etc.) [15]. Each ECU is connected to different sensors and actuators. ECU reads the sensor values at certain intervals, process them, and controls outputs/actuators. ECU also communicates the sensor values over the controller area network, CAN bus to other ECUs. Each ECU has many tasks to complete before the tasks' deadlines. The students select some of these ECUs and choose multiple task's arrival time, execution time, dead line, and different task scheduling algorithms and try to simulate using the simulators. They check whether the tasks meet the deadlines. They also try to test their scheduling algorithm using the low cost microcontroller boards.

Security and integrity of data in the automotive bus is important. The Controller Area Network Flexible Data protocol (CAN FD) supports data frame up to 64 bytes compared with 8 bytes of frame in CAN protocol. The CAN FD bus is suitable for transmitting encrypted data. The students model the encryption of automotive data using AES-128 algorithm (to provide confidentiality) and SHA-1 symmetric key (to provide integrity and authentication) [16].

Student Participation in Experiments

The real time systems course students perform a number of simulation exercises and also develop small projects with microcontroller boards (example: Traffic controller, bottling plant controller etc.). These simulations and projects help them learn the task scheduling, feasibility analysis, and implementation. The lab exercises promote active learning as against passive learning and also increases creative thinking and productive thinking [17, 18]. Students give oral presentation on their project and submit a detailed project report.

Scheduling Algorithms Analysis Tool

A software tool to select algorithm from a list for a given set of tasks and display scheduling timing scheme has been developed by students. This is to show our understanding of Priority based scheduling algorithms in a Uniprocessor system. The tool takes 'n' number of tasks and their corresponding periodicities, execution times and deadlines as inputs and provides the scheduling construct timing diagram based on the algorithm selected. Figure 3 shows a Screen shot for the task scheduling.

Figure 4 shows the use of Excel's conditional formatting to manually draw timing diagram for various tasks based on various task scheduling algorithms.

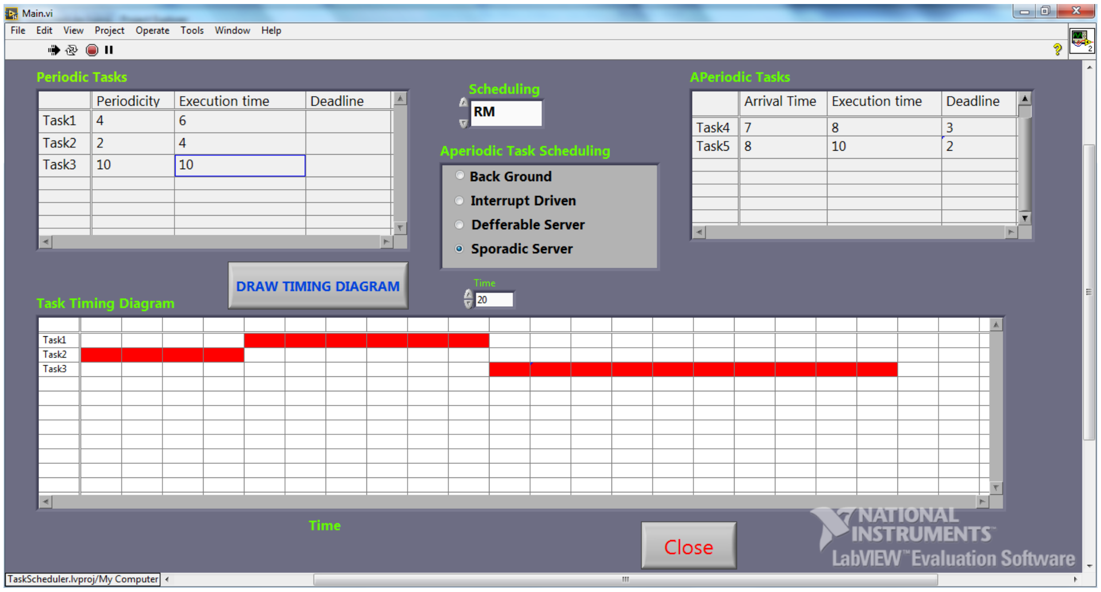

Figure 3. Screen shot for the task scheduling.

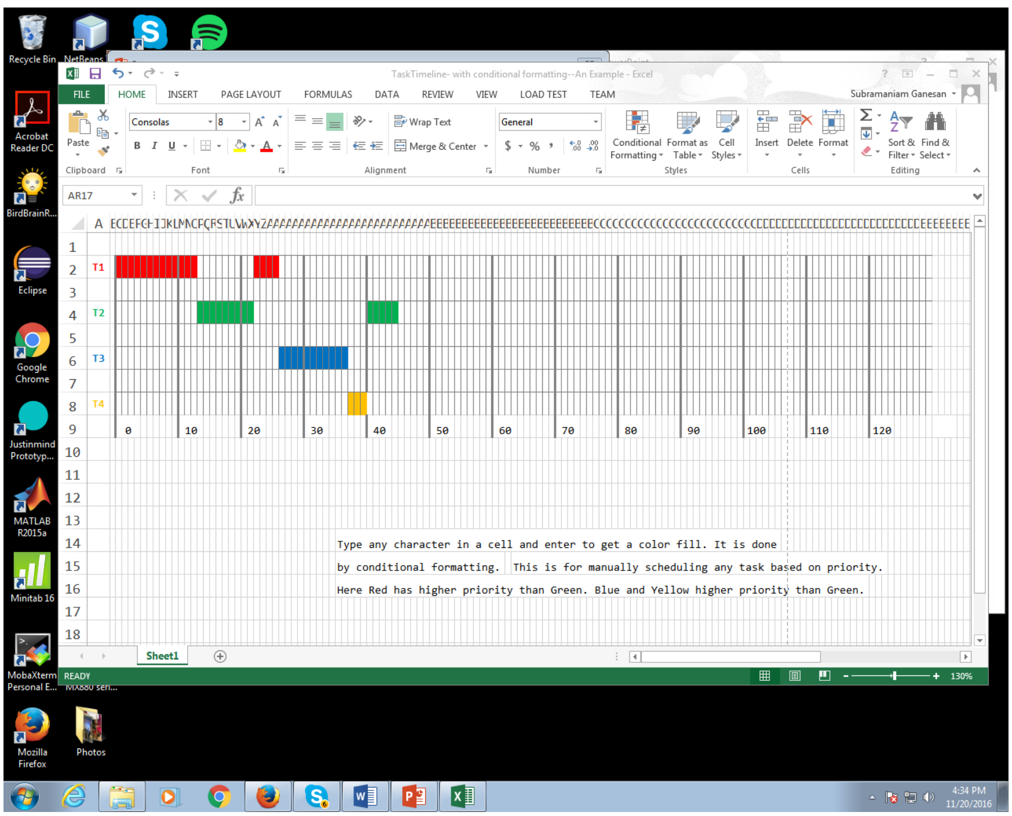

Figure 4: Excel conditional formatting cell color to draw task scheduling.

Course Objectives assessment by faculty

To make sure that the students meet the objectives of the course, the faculty considers the following points [19].

- Breakdown the objectives for the course to Weekly course objectives.
- Teach the topics to meet the weekly course objectives.
- Assign homework and quiz every week that can be used to measure objectives for every week.
- Student's abilities and skills may differ depending on their background (Example: Computer Science or Computer Engineering student). Make sure the necessary prerequisite material is covered quickly at the beginning of the semester.
- Develop and teach course material every week to challenge indirectly or directly the problem solving skills and critical thinking skills of the students.
- Students' achievements are monitored by Homework, group discussion in the class and weekly quiz so that they meet all the weekly objectives of the course.

Student Assessment of Objectives

At the end of the semester, students fill a course evaluation. Appendix B shows the end of the semester Course objective evaluation student survey questions. Anonymous survey answers are collected through the web. All students taking the course need to answer it. They rate each objective from 1 to 5 where 5 is excellent. Average and standard deviation are computed. If the average is less than 4, the instructor is expected to respond how improvements will be done to get higher rating for that objective. Appendix C shows an example survey result. An example instructor response to a situation where the average rating is below 4 is also shown. Appendix C shows also graduate program outcome evaluation (in this course) by peer review (selected faculty other than the instructor who have knowledge of this topic area). It is based on the course material, homework question and answers, exam questions and answers by students, project and lab reports. During peer review specific suggestions on the selection of objectives, material taught in the course, their emphasis/ depth and relevance to the objectives of the course are given to the teaching faculty.

Conclusions and how the Students Benefit from this Approach

The simulation exercises, lab exercises and projects have made the course material along with task scheduling algorithms easy to understand and fun to learn. The end of semester course evaluations on the course objectives received good ratings.

The topics taught in this course are useful for the automotive embedded system design and testing. This paper provides an overview of simple simulation and lab experiments to teach task scheduling algorithms for real time applications. This paper describes also how the course objectives are assessed.

References:

1. Phillip A. Laplante, "Real time systems design and analysis", 4th edition, Wiley InerScience, ISBN 978-0-470-76864-8
2. Jane W. Liu "Real Time Systems" Prentice Hall, 2000, ISBN: 0-13-099651-3.
3. C.M. Krishna and R.G. Shin, "Real time system" McGraw Hill 1997.
4. Micro C/OS-II, The real time kernel, A complete portable, ROMable, scalable preemptive RTOS by Jean J. Labrosse, R&D books, Miller Freeman Inc., ISBN: 0-87930-543-6; Phone: 785 841 1631.
5. Embedded Systems Building Blocks, 2nd edition, Complete and ready to use modules in C, by Jean J. Labrosse, R&D books, Miller Freeman Inc., ISBN: 0-87930-604-1; Phone: 1-800-788-3123.
6. Jeffrey Tsai and Steve Yang, " Monitoring and Debugging of Distributed real time system" IEEE computer Society press, ISBN 0-8186-6537-8
7. Subra Ganesan, "Teaching Real time system schedulability using low cost microprocessor board" Proceeding of ASEE NCS conference 2016.
8. Valvano, " Embedded microcontroller system- real time interfacing" Brooks/Cole publisher
9. https://cpuss.codeplex.com/
10. http://beru.univ-brest.fr/~singhoff/cheddar/
11. http://www.etas.com/en/products/ascet_software_products.php
12. http://www.st.com/web/en/news/n3784 and http://www.keil.com/arm/mdk.asp
13. http://www.arm.com/support/university/
14. Embedded Systems: Real-Time Interfacing to Arm® Cortex(TM)-M Microcontrollers 2nd Edition by Jonathan W. Valvano , July 2015, http://users.ece.utexas.edu/~valvano
15. Ronald Jurgen "Automotive Handbook", McGraw-Hill Handbook, second edition.
16. Tri Doan, Subra Ganesan, " CAN crypto FPGA chips to secure data transmitted through CAN-FD bus using AES 128 and SHA 1 with A symmetric key" SAE world congress conference, April 2017
17. http://ericbrown.com/critical-thinking-vs-creative-thinking.htm
18. Think Better: An Innovator's Guide to Productive Thinking by Tim Hurson, McGraw Hills companies, ISBN-13: 978-0-07-149493-9
19. Subramaniam Ganesan, A Rusek, D. Aloi, "An outcome-driven on-line graduate course: Real time systems course", Proceedings of ASEE NCS 2011 conference.

Appendix A: Real Time Systems Course details

Text book: Phillip A. Laplante, "Real time systems design and analysis", 4th edition,
Wiley InterScience, ISBN 978-0-470-76864-8
Reference books:
Jane W. Liu "Real Time Systems" Prentice Hall, 2000, ISBN: 0-13-099651-3.
C.M. Krishna and R.G. Shin, "Real time system" McGraw Hill 1997.

COURSE Topics:

1) Introduction and Basic concepts (1 hr.)
2) Characterization of Real time systems and Tasks (2 hr.)
    Reference model of Real Time Systems
3) Task Assignments and Scheduling (6 hr.)
    Real time scheduling
    Clock-driven scheduling
    Priority driven Scheduling
    Multiprocessor Scheduling
4) Real Time Tools
    Real time Java, Matlab Real Time Tool Box, RTAI from ETAS (1 hr.)
5) Real time Operating Systems
    RT kernels, Inter-task communication and Synchronization
    Memory Management
    Case Study: Win CE, Real time Linux, QNX
    UC/OS II real time kernel (8 hrs.)
6) Real time Communications (4 hrs.)
7) Distributed real time systems (4 hours)
8) Fault Tolerance (1 hr.)
9) Real time DSP System
    Code composer Studio
    Development Platform (6 hrs.)
10) Real Time System Development, Code Warrior software (1 hr.)
11) Performance Analysis and optimization (3 hrs.)
12) Divisible Load Theory
11) Automotive Applications (4 h)

Appendix B— End of the semester Course objective evaluation student survey— Anonymous survey- all students need to answer it.

Real-Time Computing Systems
Fall Semester 2014
Subramaniam Ganesan

Welcome to the SECS course evaluation survey on Course Objective
Course Objectives

Please rate on a scale of 5-0 how well the course taught you to meet each of the following objectives.

Objective	5 Excellent	4 Good	3 Average	2 Poor	1 Unsatisfactory	0 NA
Know the definition and characteristics of Real Time systems	○	○	○	○	○	○
Know the various task assignment and scheduling methods. For example, RM and EDF scheduling	○	○	○	○	○	○
Become familiar with Real Time system development tools like Matlab RT tool box, ETAS tools	○	○	○	○	○	○

Know the important characteristics of Real Time Operating System	○	○	○	○	○	○
Know about the RT System requirement, design and performance analysis.	○	○	○	○	○	○

Appendix C

Course Information

42284: ECE 666 Real-Time Computing Systems
Instructor : xx
Course Dates : 9/3/2014 12/13/2014
Evaluation Window : 11/27/2014 12/6/2014
Course Text: Phillip A. Laplante, “Real time systems design and analysis”, 4th edition, Wiley InerScience, ISBN 978-0-470-76864-8 Reference books: Jane W. Liu “Real Time Systems” Prentice Hall, 2000, ISBN: 0-13-099651-3. C.M. Krishna and R.G. Shin, “Real time system” McGraw Hill 1997. PREREQUISITES: CSE/ECE-570 or equivalent.

Course Objectives - Comments and Change / Delete Requests

Objective	Comment	Action
1. Know the definition and characteristics of Real Time systems Median: 4 Std. Dev: 0.7 Avg. Grade: 4.3		
2. Know the various task assignment and scheduling methods. For example, RM and EDF scheduling Median: 4 Std. Dev: 0.7 Avg. Grade: 4.3		
3. Become familiar with Real Time system development tools like Matlab RT tool box, ETAS tools Median: 4 Std. Dev: 1.1 Avg. Grade: 3.6	The move to New Building caused us to loose the License file for ETAS tools, The computer Technologist refused to change/add the license files after we moved. Next semester we do not have this problem.	
4. Know the important characteristics of Real Time Operating System Median: 4 Std. Dev: 1.1 Avg. Grade: 4.1		
5. Know about the RT System requirement, design and performance analysis. Median: 4		

Std. Dev: 1.1 Avg. Grade: 4.1		

Program Outcomes – Evaluation by Colleagues and Peers

1. Applies to: ALL
Outcome: a) An ability to apply knowledge of mathematics, science, and engineering
Status: Reinforced
no comment

2. Applies to: ALL
Outcome: b) An ability to design and conduct experiments, as well as to analyze and interpret data
Status: Reinforced
no comment

3. Applies to: ALL
Outcome: c) An ability to design a system, component, or process to meet desired needs within realistic constraints such as economic, environmental, social, political, ethical, health and safety, manufacturability, and sustainability.
Status: Reinforced
no comment

4. Applies to: ALL
Outcome: d) An ability to function on multi-disciplinary teams
Status: Reinforced
no comment

5. Applies to: ALL
Outcome: e) An ability to identify, formulate, and solve engineering problems
Status: Reinforced
no comment

6. Applies to: ALL
Outcome: f) An understanding of professional ethical responsibility
Status: Reinforced
no comment

7. Applies to: ALL
Outcome: g) An ability to communicate effectively
Status: Reinforced
no comment

8. Applies to: ALL
Outcome: h) The broad education necessary to understand the impact of engineering solutions in a global and societal context
Status: Reinforced
no comment

9. Applies to: ALL
Outcome: i) A recognition of the need for, and an ability to engage in, life-long learning
Status: Reinforced
no comment

10. Applies to: ALL
Outcome: j) A knowledge of contemporary issues
Status: Reinforced
no comment

11. Applies to: ALL
Outcome: k) An ability to use the techniques, skills, and modern engineering tools necessary for engineering

practice
Status: Reinforced
no comment