ECE 350
Laboratory Project Manual for Real-Time Operating Systems

by

Yiqing Huang
Seyed Majid Zahedi

Electrical and Computer Engineering Department
University of Waterloo

Waterloo, Ontario, Canada, January 11, 2021

Contents

List of Tables

List of Figures

Preface

Who Should Read This Lab Manual?

This lab manual is written for students who will design and implement a small Real-Time Executive (RTX) for Intel DE1-SoC board populated with a Cyclone V SoC chip, which has a dual-core ARM Cortex-A9 Hard Processor System (HPS) and Altera FPGA.

What is in This Lab Manual?

The first purpose of this document is to provide the descriptions and notes for the laboratory project. The second purpose of this document is a quick reference guide of the relevant development tools for completing laboratory projects. This manual is divided into three parts.

Part I describes the lab administration policies.

Part II is the project description. We break the project into the following five laboratory projects.

- P1: Introduction to Kernel Programming and Memory Management
- P2: Task Management
- P3: Inter-task Communications and Console I/O
- P4: Timing Service and Real-Time Scheduling
- P5: TBD

Part III is frequently asked questions.

Part IV introduces the computing environment and the development tools. It includes a DE1-SoC hardware and software reference guide. The topics are as follows.

- Windows 10 Remote Desktop
- Software Development Environment

- DE1-SoC Hardware Environment
- Programming DE1-SoC

Acknowledgements

Our project is inspired by the original ECE354 RTX course project created by Professor Paul Dasiewicz. Professor Dasiewicz provided detailed notes and sample code to us. We sincerely thank the following generous donations, without which the lab will not be possible:

- ARM University Program for providing us with lab teaching materials and ARM DS gold edition software licenses.
- Intel University Program for providing us with 50 DE1-SoC FPGA boards.
- TerasIC, the manufacture, for shipping the boards in a timely manner.
- Imperas Software for providing us one evaluation license to experiment with their software tools during the lab development.

We gratefully thank our graduate teaching assistants: Ali H. A. Abyaneh, Weitian Xing, and Maizi Liao for their strong support of the lab including developing part of the lab test cases. Our gratitude also goes out to Eric Praetzel for his continuous strong support of the IT infrastructure of RTOS lab hardware and the ARM DS software, Rasoul Keshavarzi-Valdani for lending us the a DE1-SoC board to experiment with during the initial board selection phase of the lab development. Kim Pope and Reinier Torres Labrada both provided helpful FPGA tips and we gratefully acknowledge their expertise and help.

Finally we own many thanks to our students who did ECE354, SE350 and ECE350 course projects in the past and provided constructive feedback. The lab projects won't exist without our students.

Part I

Lab Administration

Lab Administration Policy

Group Lab Policy

- **Group Size.** All labs are done in groups of *four*. A group size of less than four is not recommended. There is no reduction in project deliverables regardless the size of the project group. The Learn system (http://learn.uwaterloo.ca) is used to signup for groups. The lab group sign-up deadline is in Table 0.1). Late group sign-up is not accepted and will result in losing the entire lab sign-up mark, which is **2%** of the total lab project grade. Grace days do not apply to Group Sign-up. Any student without a lab group after the sign-up deadline will be randomly assigned to a lab group by the lab teaching staff.

- **Group Project Manager.** The group elects one member as the group project manager. The project manager can be the same person for all deliverables or a different person for a different deliverable. Rotating project manager's role gives each group member an opportunity to practice group project management. However this role rotation is a choice rather than requirement. It is up to the group to decide. You need to submit the group information in .csv file every time there is an update of the project manager or group membership. A `group.csv` template file can be found at https://github.com/yqh/ece350 under the submission sub-directory.

- **Quitting from a Group.** If you notice workload imbalance, try to solve it as soon as possible within your group. Quitting from the group should be used as the last resort. Group quitting is only allowed once. You are allowed to join another group which has three or less number of students. You are not allowed to quit from the newly formed group again. There is *one grace day deduction penalty* to be applied to each member in the old group. We highly recommend everyone to stay with your group members as much as possible, for the ability to do team work will be an important skill in your future career. Please choose your lab partners carefully and wisely. The code and documentation completed before the group split-up are the intellectual property of each students in the old group.

- **Group Quitting Deadline.** To quit from your group, you need to notify the lab instructor in writing and sign the group split-up form (see the Appendix A) at

Deliverable	Weight	Due Date	File Name
P0 Group Sign-up	2%	16:30 EST Jan 14	group.csv
P1 Memory Management	18%	16:30 EST Jan 28	p1_Gid.zip
P2 Task Management	20%	16:30 EST Feb 11	p2_Gid.zip
P3 Inter-task Communications and Console I/O	25%	16:30 EST Mar 11	p3_Gid.zip
P4 Timing and Real-Time Scheduling	20%	16:30 EST Mar 25	p4_Gid.zip
P5 TBD	15%	16:30 EST Apr 08	p5_Gid.zip

Table 0.1: Project Deliverable Weight and Deadlines. Replace the "id" in "Gid" with the two digit group ID number.

least one week before the nearest lab project deadline.

Project Submission Policy.

- **Project Deliverables.** The lab project is divided into five deliverables. For each deliverable, there is a pre-lab deliverable and a post-lab deliverable. Students are required to finish the pre-lab deliverable before attempting the lab assignments. For the terms we have scheduled lab sessions, pre-lab is due by the time your scheduled lab session starts. For the terms we do not have scheduled lab sessions, pre-lab is due by the deadline of the previous lab's post-lab.

  Each post-lab deliverable includes the source code, a lab report (in pdf) file and the `group.csv` [1]. Create a directory and name it "labN", where N is 1, 2, ..., 5. Create a sub-directory named "code". Put your ARM DS application folder under the code directory. Name your lab report file "pN_report.pdf", where N is 1, 2, 3, 4, 5 and put it under labN directory. Include a README file with group identification, project manager name and a description of directory contents. Put the README file under labN directory. Archive all files for each deliverable in a single file and submit it to the corresponding Learn Dropbox. Table 0.1 gives the weight, deadline and naming convention of each post-lab deliverable.

  For each deliverable, we will conduct an anonymous peer review within a group. A student will rate how satisfied he/she is with every other group member's contribution from 0 to 10, where the higher the rating, the more satisfied the student feels about the contribution the other member has done for the project. This is to make sure everyone in the group will contribute their fair share to the project. We will use simple arithmetic average ratings each

[1] You are required to submit the group.csv whenever there is an update of your group membership or project manager role. When there is no update, you are welcome to submit it again, though not required.

Peer Rating	Contribution Factor CF
[7, 10]	100%
[6, 7)	80%
[5, 6)	60%
[4, 5)	40%
[0, 4)	0%

Table 0.2: Group Project contribution factor table. Each student's lab grade is their group project grade multiplied by the CF (Contribution Factor).

group member received and assign individual lab grade to each team member by multiplying the project grade with a contribution percentage factor listed in Table 0.2. The peer review submission deadline is one hour after the project deadline or one hour after the project submission if the project submission is after the project deadline.

- **Late Submissions**

  Late submission is accepted within three days after the deadline. Please be advised that late submission is counted in a unit of day rather than hour or minute. An hour late submission is one day late, so does a fifteen hour late submission. Unless notified otherwise, we always take the latest submission from the Learn dropbox. Assume the submission is H hours after the deadline. Then the number of days you are late is computed by the following function given the hours your are late:

```
#include <math.h>

int get_late_days(double late_hours) {
   return (int) (ceil(late_hours/24));
}
```

  - **Late Project Submissions.** There are *five grace days*[2] that can be used for project deliverables late submissions without incurring any penalty. A group split-up will consume one grace day. When you use up all your grace days, a **15%** per day late penalty will be applied to a late submission. *Submission is not accepted if it is more than three days late.*
  - **Late Peer Review Submissions.** Late peer review submission is accepted within three days after the peer review submission deadline. A **5%** per day late penalty will be applied to individual's lab grade when the peer review submission is late.

[2]Grace days are calendar days. Days in weekends are counted.

Project Grading Policy

- **Project Grading Procedure.** The project is graded by automated testing framework. For each deliverable, we publish a small set of testing cases. We require students to pass these testing cases before they submit. If you are not able to pass these testing cases, then you will be given a chance to demo your project and the maximum grade you will get would be capped to 70.

- **Hardware vs. Simulator.** Submissions will be evaluated on a Windows 10 lab machine that has a board attached to it. Lab machines are accessible through ENGLab remote desktop session when connected to the campus virtual private network (VPN). If a lab requires the program to run on the board, but the program only functions inside the simulator, a **15%** penalty will be applied the particular lab's grade.

- **Project Demo Policy.** Every group will have a chance to request a demo session with a grading TA after lab grades are released. The purpose of project demo is mainly for re-grading your project. If you have no concerns with your lab grade, then you are not required to attend the demo. Each demo has a time limit. You are allowed to make up to 1024 bytes of change of the source. Any change of the source code will consume one grace day or a 15% late submission penalty should all grace days are exhausted. Note change of the source code needs to be done through a text editor. Directly cut and paste from a file not within your submission or replace a source file with an file not in your submission is strictly prohibited.

  If you are not interested in re-grading your project, but want to ask grading TA some questions or advises, you may also request a demo after the grades are released.

- **Second Project Re-grading after the Demo.** If you are still not satisfied with the grades received after the demo, escalate your case to the lab instructor to request a review and the lab instructor will finalize the case. Please note any re-grading (including the demo re-grading) is a rigid process. The entire project is subject to be re-graded. Your new grades may be lower, unchanged or higher than the original grade received.

Lab Repeating Policy

For a student who repeats the course, labs need to be re-done with new lab partners. Simply turning in the old lab code is not allowed. We understand that the student may choose a similar route to the solution chosen last time the course was taken. However it should not be identical. The labs will be done a second time, we expect that the student will improve the older solutions. Also the new lab partners should be contributing equally, which will also lead to differences in the solutions.

Note that the policy is course specific to the discretion of the course instructor and the lab instructor.

Lab Projects Solution Internet Policy

Publishing your lab projects solution source code or lab report on the internet for public to access is a violation of academic integrity. Because this potentially enabling other groups to cheat the system in the current and future offerings of the course. For example, it is not acceptable to host a public repository on GitHub that contains your lab project solutions. A lab grade zero will automatically be assigned to the offender.

Seeking Help

- **Discussion Forum.** We recommend students to use the Piazza discussion forum to ask the teaching team questions instead of sending individual emails to lab teaching staff. For questions related to lab projects, our target response time is one business day before the deadline of the particular lab in question [3]. *There is no guarantee on the response time to questions of a lab that passes the submission deadline*.

- **Office Hours.** The lab office hours are for group project consultation. Your entire group may attend the same appointment. Each appointment is a 15 minute time slot. Book multiple consecutive time slots if you need more time. All appointments require a minimum 1 hour lead time. The maximum lead time you can book a lab office hour is 5 days. You should cancel your booked appointment if you are not able to attend it. There are other students that may need to slots, so please do not book appointments if you are not able to make the appointment.

- **Appointments.** Students can also make appointments with lab teaching staff should their problems are not resolved by discussion forum or during office hours. The appointment booking is by email.

  To make the appointment efficient and effective, when requesting an appointment, please specify three preferred times and roughly how long the appointment needs to be. On average, an appointment is fifteen minutes per project group. Please also summarize the main questions to be asked in your appointment requesting email. If a question requires teaching staff to look at a code fragment, please make sure your code is accessible by the lab teaching staff.

[3] Our past experiences show that the number of questions spike when deadline is close.The teaching staff will not be able to guarantee one business day response time when workload is above average, though we always try our best to provide timely response.

Please note that teaching staff will not debug student's program for the student. Debugging is part of the exercise of finishing a programming assignment. Teaching staff will be able to demonstrate how to use the debugger and provide case specific debugging tips. Teaching staff will not give direct solution to a lab assignment. Guidances and hints will be provided to help students to find the solution by themselves.

Part II

Lab Project

Chapter 1

Introduction

1.1 Overview

In this project, you will design a small real-time executive (RTX) and implement it on a Intel DE1-SoC board populated with the Cyclone V SoC chip, which has a dual-core ARM Cortex-A9 Hard Processor System (HPS) and Altera FPGA. The executive will provide a basic multiprogramming environment, with five priority levels, preemption, dynamic memory management, mailbox for inter-task communications and synchronization, a basic timing service, system console I/O and debugging support, and finally real-time scheduling.

Such an RTX is suitable for embedded computers which operate in real time. A cooperative, non-malicious software environment is assumed. The design of the RTX should allow its placement in ROM. Unprivileged RTX tasks must execute under the user mode of the Cortex-A9 processor. The RTX kernel and kernel tasks will execute in the privileged level under supervisor mode.

On the HPS side, there is an on-chip RAM of 64 KB and a DDR3 RAM of 1 GB for use by the RTX and application tasks. The board has four Hard Processor System (HPS) timers, two JTAG UARTs and several other peripheral interface devices. The UART0 is used for your RTX system console and UART1 is used for your RTX debug terminal.

1.2 Summary of RTX Requirements

The RTX requirements are listed as follows:

1.2.1 RTX Primitives and Services

The RTX provides primitives and services for as following.

Memory Management

First fit dynamic memory allocation is supported. Refer to Chapter 2 for details.

Task Management

The RTX fixed number of tasks. The maximum number of tasks that can run is decided at compile time. The RTX supports task creation and deletion during run time. The RTX is supports task preemption. There are four user priority levels plus an additional "hidden" priority level for the Null task. There is no time slicing. FIFO (First In, First Out) scheduling policy at each priority level is supported. Refer to Chapter 3 for details.

Synchronization, Timing Service and Console I/O

The RTX provides mailbox utility for inter-task communication and synchronization. An interrupt-driven UART provides the console service. The RTX provides a primitive for a task to pause itself and for a primitive to query the kernel internal clock ticks. Refer to Chapter 4 for details.

Real-Time Dynamic Scheduling

The EDF (Earliest Deadline First) scheduling policy at each priority level is supported. Refer to Chapter 5 for details.

1.2.2 RTX Tasks

You are required to implement two types of tasks by using the RTX primitives and services. They are user tasks and kernel tasks.

User Tasks

These tasks are operating at a unprivileged level in user mode. They are user applications that perform certain user defined functions. For each lab project, you will implement test tasks to help you test the RTX primitives and services you have designed and implemented. In later labs, you will add tasks that require console I/O services once you have the console I/O service ready.

System Tasks

These tasks are operating in user mode or supervisor mode. Some may require a privileged level of operation and some may be sufficient to operate at a unprivi-

leged level. It is your design decision to justify which task will be operating at what privilege level. Three system tasks are required and they are null task (see Chapter 2), console display task and keyboard command decoder task (see Chapter 4).

1.2.3 RTX Footprint and Processor Loading

A reasonably *lean* implementation is expected. No standard C library function call is allowed in the kernel code.

1.2.4 Error Detection and Recovery

The primitive will return an error code (a non-zero integer value) upon an error. No error recovery is required. It may be assumed that the application processes can deal with this situation.

Chapter 2

Lab1

To be released.

Chapter 3

Lab2

To be released.

Chapter 4

Lab3

To be released.

Chapter 5

Lab4

To be released.

Chapter 6

Lab5

To be released.

Part III

Frequently Asked Questions

Part IV

Computing and Software Development Environment Quick Reference Guide

Appendix A

Forms

Lab administration related forms are given in this appendix.

ECE 350 Request to Leave a Project Group Form

Name	
Quest ID	
Student ID	
Lab Project ID	
Group ID	
Name of Other Group Member 1	
Name of Other Group Member 2	
Name of Other Group Member 3	

Provide the reason for leaving the project group here:

Signature _____ Date _____

Bibliography