LEARNING

GUIDE

Faculty of Engineering and Information Technology

School of Electrical, Mechanical and Mechatronic Systems

48434
Embedded Software

Autumn 2017

SUBJECT GUIDE

Welcome

This subject focuses on developing a set of field-of-practice skills and knowledge:

- It develops an applied base for your field-of-practice knowledge.
- It develops competence in the use of software development tools through laboratory-based project work and problem-based learning.
- It lets you apply core skills and knowledge to a field-of-practice.

Embedded Software is a subject in the last year of the course – it assumes that you have developed a proficiency in academic and information literacy skills, and provides specialised knowledge for a part of your field-of-practice. It also helps you to apply that specialist knowledge to practical, real-world problems in a laboratory setting and prepares you for the graduate workplace.

You will be expected to take on a significant responsibility for your own learning. While self-managed learning offers you choices about how and when you study, we also understand that you will learn best if there are convenient opportunities for you to interact with fellow students and staff.

Therefore, the subject provides a balance between the convenience of independent learning and the stimulation of academic life. We hope you enjoy the content, learning experiences and assessment tasks that make up this subject as well as the benefits of managing your own learning.

Your Subject Coordinator

Dr Peter McLean is the Deputy Head of School (Teaching & Learning) in the School of Electrical, Mechanical and Mechatronic Systems within the Faculty of Engineering and Information Technology. Subjects taught include Electronics and Circuits, Introductory Digital Systems, Fundamentals of Electrical Engineering, Circuit Analysis, Signals and Systems, Data Acquisition and Distribution, Digital Electronics, Analog Electronics, Signal Processing, Embedded Software, Power Circuit Theory and Power Systems Operation and Protection.

He has undertaken numerous research projects in collaboration with industry that normally involve the development of embedded systems hardware and software. These include microcontroller-based power system protection devices, DSP-based power-line carrier systems and a broadband Internet distribution system for the home.

Where this subject fits into the course

This subject is a Stage 7 field-of-practice subject in the Embedded Systems thread which is a part of the Electrical Major and ICT / Software Major within various Bachelor of Engineering Degrees.

The need for this subject

It is assumed that you have already been introduced to and attained competence in the C programming language, digital logic design fundamentals, and learnt to design simple sequential programs for a device other than a PC. In this subject you will gain experience in the design of software for an embedded application. It will be seen that embedded software draws on many fields of engineering expertise, and that techniques of synthesis are highly dependent on system specifications.

The subject lays the foundation for many areas of further interest to the engineer – autonomous systems and robotics, software architecture, real-time operating systems, signal processing and data engineering.

Subject aims and objectives

The objective of this subject is for you to design and test software for an embedded system.

You will bring together many elements of engineering – system specification, design, simulation, testing and management – all in the context of a high level of integration between electronics and software.

The technical content of the subject aims to develop the basic structure, design and operation of embedded systems from the software perspective. The subject will give you practice in designing software for real embedded systems.

Skills in writing software, interfacing, debugging and experimental verification are developed through a series of laboratories. A project in which you analyse, design, implement and test part of an embedded system contextualises nearly all the technical content and makes use of the previously acquired skills.

"The artist is nothing without the gift, but the gift is nothing without work. "
- Emile Zola (1840-1902)

Three engineering themes permeate the subject. The first theme is the need for a systems perspective in engineering – you will need to analyse and dissect (through a requirements specification) and eventually synthesise in a hierarchical manner (through software design). The second theme is that you will be expected to draw knowledge from a wide variety of sources – previous subjects, industrial experience, industry-produced datasheets and application notes and the Internet. The third theme is that of the need for engineers to take responsibility for their own professional development. You will be responsible for your own learning – which will encompass requirements specifications, mathematical modelling, electronics interfacing, software design and testing.

Finally, the subject will prepare you for more advanced topics on software systems, operating systems and signal processing which you may encounter in professional practice and in further subjects.

Other subject information

The following information takes precedence over the default policies outlined in section 3.3.1 of the Faculty's Student Guide.

Laboratories

face to face session

The laboratories are structured sessions that allow you to put into practice the knowledge delivered in the Notes, using specialised equipment.

Twenty-four hour access to the Embedded Systems Laboratory will be given to students during the semester.

You should attend all the laboratories. During the laboratory sessions you will have the opportunity to meet with fellow students and with your subject coordinator who will answer questions and highlight selected topics.

Safety first!

As part of the Faculty's commitment to safety, all students are now required to complete a safety induction. Access to laboratories is dependent upon your successful completion of the safety induction. The induction must be renewed each year.

Laboratory access is contingent upon successful completion of the UTSOnline Safety Induction Quiz.

Software

A cross-compiler for the C language and an associated toolchain for the Kinetis series of microcontrollers will be used extensively throughout the subject as a means of programming an embedded system.

The computer laboratories have a freeware version of NXP's Kinetis Design Studio Integrated Development Environment which you may wish to obtain.

The subject will rely heavily on a freely available Version Control System and Windows Explorer shell extension called TortoiseSVN.

Assessment

Assessment for this subject is criterion-referenced. This means that your performance is measured against a set of criteria, not against the performance of other students.

The assessment criteria for this subject

In assessing your performance we will be looking for evidence that:

- You are able to efficiently carry out an accurate analysis of the requirements of embedded software which are similar to those dealt with in this subject.
- You have understood the concepts used in embedded software design and are able to apply them to the design of practical systems such as simple embedded systems.
- You can distinguish between the different methods of implementing real-time embedded software and know their limitations and how to apply them correctly.
- You have understood the methods of hardware interfacing and are able to apply them to practical embedded systems such as those covered in this subject.

"Not everything that can be counted counts, and not everything that counts can be counted." - Albert Einstein (1879-1955)

Calculators

Programmable calculators **are** allowed for the quiz.

STUDY GUIDE

There are three components to completing your study of Embedded Software. They are:

- studying the Notes and associated readings
- attempting the laboratory assessment tasks
- completing the project satisfactorily

To guide you through these tasks there is a Timetable.

Structure of the Timetable

The Timetable will help you manage your learning in Embedded Software. It does so through the following design features:

- It is organised in logical, linked and digestible steps, so that where your learning is headed remains clear. Each program activity of the Timetable refers to:
  - A topic in the Notes. Each topic may have associated readings that should be studied in preparation for the practical laboratory tasks.
  - Assessment tasks that are due.

- The Timetable asks you, therefore, to be an active learner; not a passive reader. You should keep in mind that to achieve the necessary competence to pass this subject it is not sufficient to just read the pages of the Notes and readings a few weeks before assessment tasks are due. Apart from understanding the concepts given in the Notes, you also need to practice writing and debugging software and allow yourself sufficient time to reflect on what you have learnt.

- You can see what the learning tasks will be for each program activity of the Timetable before you begin. This enables you to mentally prepare for the learning tasks while you work through the program activity topic. In this way your learning stays focused on the main areas of the program activity; you don’t lose your way in the details.

The Notes

The Notes should be read before each program activity so that face-to-face time can be used to clarify or expand on particular topics of interest.

Structure of the Notes

The Notes summarize important topics and complement the readings. They are updated regularly as technology changes, and from feedback from students attempting to learn the topics. Difficult or hard-to-grasp topics are expanded; or are presented in a different manner to the readings; or highlight the real-world application of the topic. Prerequisite material is often recapped. The focus of the Notes is towards the final project, so those topics that are important to this goal are treated fairly thoroughly.

Skim through first

Margin notes help you navigate the material

If you are already familiar with the material in any section or if you want to get an overall feel for what it contains, you may like to skim through it first, looking at the headings and margin notes.

Learning in partnership

Using a fellow student as a learning partner has been found repeatedly to be an important learning support. The idea is that you contact a fellow student, by whatever means is most convenient, to discuss your interpretation of a learning task, to check if your approaches are the same and to generally clear up any confusions which may have arisen. It has been found that well over half of the concerns students experience about their learning are to do with simply checking that they are 'on the right track' and can be solved using this method. If, however, the concern or uncertainty remains, it is then recommended that you contact your Subject Coordinator.

Your learning plan

Organising your time is a major challenge in learning. Leaving recommended readings and assessment tasks to the last minute is a common problem. To assist you with this challenge you may find it useful to plan your study time before you start work on this subject. First decide on the best place and time each week to study without distractions and then make sure to adhere to your own plan.

It is estimated that you should set aside a total of approximately 9 hours of study time each week. It is recommended that you break up those hours into at least two study sessions each on a different day of the week. This is a rough guide only, as people learn at different rates and from different levels of experience.

Timetable

DATE	NOTES	READINGS	ASSESSMENT
1A 13 Mar	***1 Embedded Systems*** Overview of Embedded Systems. Overview of Tower board. NXP K70F120M architecture. Lab safety. Equipment familiarisation. Kinetis Design Studio.	http://cache.freescale.com/files/microcontrollers/doc/user_guide/TWRK70F120MUM.pdf http://cache.freescale.com/files/microcontrollers/doc/ref_manual/K70P256M150SF3RM.pdf http://cache.freescale.com/files/32bit/hardware_tools/schematics/TWR-K70F120M-SCH.pdf https://community.freescale.com/community/kinetis/kinetis-design-studio/content?filterID=contentstatus%5Bpublished%5D~objecttype~objecttype%5Bvideo%5D	
1B 15 Mar	***2 Embedded C*** Review of the C language. Initializing and accessing I/O ports. Memory allocation. Self-documenting code. Modular software development. Layered software systems. Debugging.	Chapter 10 of K70P256M150SF3RM.pdf Chapter 11 of K70P256M150SF3RM.pdf Lab 1	

DATE	NOTES	READINGS	ASSESSMENT
2A 20 Mar	***3 Microcontroller Architecture*** Clock generation and distribution. UART. PC USB Interface. FIFOs. Polling. Tower serial protocol.	Chapter 56 of K70P256M150SF3RM.pdf §56.1, 56.1.1, 56.2, 56.3.1, 56.3.2, 56.3.3, 56.3.4, 56.3.5, 56.3.8, 56.3.11, 56.4, 56.4.2, 56.4.2.1, 56.4.2.3, 56.4.3, 56.4.3.1, 56.4.3.3, 56.4.3.8, 56.4.3.8.1, 56.4.3.8.2, 56.4.4, 56.4.5, 56.4.5.1, 56.4.5.3, 56.4.5.3.1, 56.5, 56.8, 56.8.1, **56.8.3** http://cache.freescale.com/files/32bit/doc/quick_ref_guide/KQRUG.pdf Tower Serial Communication Protocol	
2B 22 Mar	Lab work.		
3A 27 Mar	***4 Memory*** Flash memory. EEPROM. RAM. Special function registers. Memory-mapped peripherals.	Chapter 29 of K70P256M150SF3RM.pdf Chapter 30 of K70P256M150SF3RM.pdf	
3B 29 Mar	Lab assessment.	Lab 2	**Lab 1**
4A 3 Apr	***5 Interrupts*** Interrupts. Interrupt service routines. Hardware interrupts. Interrupt vectors and priority. Exceptions. Threads. Foreground and background threads. Re-entrant programming.	Chapter 56 of K70P256M150SF3RM.pdf §56.6	

X

4B 5 Apr	Lab work.		

DATE	NOTES	READINGS	ASSESSMENT
5A 10 Apr	***6 Timing Generation and Measurements*** Timer module. Periodic timer. Output compare. Input capture. Pulse accumulator.	Chapter 43 of K70P256M150SF3RM.pdf Chapter 44 of K70P256M150SF3RM.pdf	
5B 12 Apr	Lab assessment.	Lab 3	**Lab 2**
6A 17 Apr	***Public Holiday***		
6B 19 Apr	Lab work		

DATE	NOTES	READINGS	ASSESSMENT
S1A 24 Apr	***7 Concurrent Software*** Threads. Schedulers. Operating systems. The semaphore. Mutual exclusion with semaphores. Synchronisation with semaphores. The producer / consumer problem with semaphores.		
S1B 26 Apr	***Mid-Session StuVac***		
7A 1 May	***8 Interfacing*** Input switches and keyboards. Analog to digital conversion. Digital to analog conversion. Inter-Integrated Circuit (I^2C).	Chapter 55 of K70P256M150SF3RM.pdf MMA8451Q Datasheet	
7B 3 May	Lab assessment.	Lab 4	**Lab 3**
8A 8 May	***9 Fixed-Point Processing*** *Q*-notation. Other notations. Fixed-point calculations. Square-root algorithm for a fixed-point processor.	Chapter 55 of K70P256M150SF3RM.pdf http://infocenter.arm.com/help/topic/com.arm.doc.dai0033a/DAI0033A_fixedpoint_appsnote.pdf	
8B 10 May	***Quiz*** Topics 1-8 inclusive.		**Quiz**

DATE	NOTES	READINGS	ASSESSMENT
9A 15 May	***10 Real-Time Operating Systems*** Real-time kernel concepts. Re-entrancy. Thread priority. Mutual Exclusion. Synchronization. Inter-thread communication. Interrupts. Memory requirements. Advantages and disadvantages of real-time operating systems.		
9B 17 May	Lab assessment.	Lab 5	**Lab 4**
10A 22 May	***Embedded Software Project*** Overview of project.	Project	
10B 24 May	***Embedded Software Project*** Project work.		
11A 29 May	***Embedded Software Project*** Project work.		
11B 31 May	Lab assessment.		**Lab 5**

DATE	NOTES	READINGS	ASSESSMENT
12A 5 Jun	***Embedded Software Project*** Project work.		
12B 7 Jun	***Embedded Software Project*** Project work.		
S2A 12 Jun	***Public Holiday***		
S2B 14 Jun	***Final StuVac***		
A1A 19 Jun			
A1B 21 Jun			
A2A 26 Jun			
A2B 28 Jun	***Embedded Software Project*** Project marking.		**Project**