EN0572: Operating systems and concurrency
Course Work 2015-16

Module tutor: David Kendall

1 Dates and mechanisms for assessment submission and feedback

Date of hand out to students: 2nd November 2015

Mechanism to be used to disseminate to students: eLP

Date and Time of Submission by Student:

- Demonstration: In lab session in w/c 7th December 2015
- Reports: 23.59 on 11th December 2015

Mechanism for Submission of Work by Student: The reports should be submitted via eLP using the Turnitin links as follows:

- Software report: `Assessment->Software Report`.
- Theory and concepts report: `Assessment->Theory Report`.

Date by which Work, Feedback and Marks will be returned to Students: Within 3 working weeks of submission date.

Mechanism(s) for return of assignment work, feedback and marks to students: eLP + appointment/email.

2 Assignment brief

2.1 Introduction

You are required to develop software that uses the uC/OS-II operating system to implement a security briefcase alarm on a LPC4088 ARM board. Your software should be written in standard C and should run on an ARM board in PB S2.

You should imagine that an LPC4088 board could be fixed to a briefcase to act as a security device. When the security device is not enabled the briefcase can be locked by pressing the joystick `UP` and unlocked by pressing the joystick `DOWN`. The security mechanism is enabled by pressing the joystick `LEFT`. It should not be possible to enable the security mechanism when the briefcase is unlocked. Once the security mechanism has been enabled, the briefcase cannot be unlocked until the mechanism is disabled. The mechanism can only be disabled by entering a pre-defined 4 digit security code (PIN number). When enabled, if the device detects any motion of the briefcase, it must be disabled within some pre-defined time interval (known as the `ALARM_INTERVAL`), starting when the motion was first detected. A count of the time remaining should be shown on the display, as should the `ALARM_INTERVAL`. If the correct code is entered in time, the security mechanism is disabled and the briefcase can be opened by pressing the joystick `DOWN`. If the correct code is not entered in time, the alarm is raised. A typical device, currently available for sale, makes a loud (100 dB) noise as its alarm. In order to avoid disruption to others in the lab, your device should simply flash the LEDs and display an appropriate status message instead. The alarm can be turned off only by entering the correct 4 digit code, following which the security mechanism is disabled. The user should use the joystick to enter the security code: `LEFT` and `RIGHT` should be used to cycle round the digits on the display; `UP` and `DOWN` should be used to increment/decrement the selected digit; and `CENTER` should be used to enter the code. The value of the `ALARM_INTERVAL` can be adjusted only when the security mechanism is disabled. It should be possible to use the red knob (potentiometer) to select an integer value between 10 and 120 seconds.

A typical display may look like this:

```
Alarm    : PENDING
Interval : 120
Time     :   7
Case     : LOCKED
           MOVING
Code     : 0 0 2 4
             -
```

2.2 Additional requirements

You should begin your solution by cloning the `git` repository `http://github.com/DavidKendall/briefcase`. Your software should make use of uC/OS-II and should comprise at least the following tasks:

- Three input tasks:
  1. buttons and joystick task
  2. potentiometer task
  3. accelerometer task
- Two output tasks:
  1. LCD task
  2. LED task

Each task should be concerned with a single well-defined aspect of the overall program functionality, e.g. the potentiometer task should just read the potentiometer value and adjust the `ALARM_INTERVAL`, if allowed to do so; it should not manipulate the LCD or the LED directly. That is the job of the output tasks. You MUST use a circular buffer protected by semaphores for all communication between tasks. You may use the `mbed` and `EALIB` libraries to access the peripheral devices: joystick, display, leds, accelerometer, potentiometer etc.

3 Systems and Concurrent Programming (50%)

3.1 Software Demonstration

You will be required to attend a demonstration in a lab session in week 12. This should be regarded as a formal University examination. Failure to attend will result in a mark of zero for the demonstration.

Marks at the demonstration will be awarded as follows:

1. Demonstrated functionality of the program *(10 marks)*
2. Quality and understanding of code *(10 marks)*

3.2 Software Report

You are required to produce a report on your software addressing the topics below.

1. *Design and concurrency*
   (a) Give a brief overview of the design of your program. Pay particular attention to your management of concurrency, e.g. you should address how concurrency is introduced into your program and how interference between tasks is avoided.
   (5 marks)
   (b) Explain any specific concurrent programming techniques that you have adopted, e.g. bounded buffer. Illustrate your answer with diagrams, if appropriate.
   (5 marks)
   (c) Briefly consider alternative approaches that you could have adopted to the design and implementation of your program and justify your chosen approach.
   (5 marks)
2. *Low-level systems programming*
   (a) Identify in your program two examples of the use of *memory-mapped I/O*. Write out the low-level code for your chosen examples and identify the relevant addresses of the memory-mapped device.

Explain thoroughly how the I/O functionality is achieved in each case. (Remember that the `mbed` library code is also part of your program.)

(6 marks)

(b) Give examples from your program of event-handling by *polling* and event-handling by *interrupts*. Explain clearly the difference between these two approaches to event-handling. Discuss the advantages and disadvantages of each approach. Illustrate your answer with examples.

(9 marks)

4 OS theory and concepts (50%)

You are required to produce a report addressing the topics below.

1. *Process management*
   Describe in detail the actions taken by an operating system to achieve a context switch between processes. Illustrate your answer with diagrams.

   (10 marks)

2. An important requirement of many operating systems is to provide a secure communication function between processes. One approach to the provision of such a function is the *Secure Socket Layer* (SSL).

   (a) Identify what potential security violations are addressed by SSL. Explain briefly how SSL offers protection against each potential violation that you identify.

   (6 marks)

   (b) SSL makes use of both *symmetric* and *public-key* cryptography. Explain what you understand by these concepts, distinguishing clearly between them. Give examples of each. Identify how each of these cryptographic techniques is used in SSL, explaining the reasons for the choice of technique in each case.

   (7 marks)

   (c) SSL is susceptible to a *man-in-the-middle* attack. Explain the nature of this attack. Identify the precise vulnerability of SSL to

this attack and discuss how users of SSL can protect themselves against it.

(7 marks)

3. *Filesystem*
   Suppose that the developers of the alarm system that you developed in your software project decide to add a file storage capability to the device so that a log can be kept of all access attempts and alarm events, and so that the configuration parameters can be stored permanently. Sensibly, they decide to make use of the SD card device of the LPC2378-STK as the storage medium, with FAT16 as the filesystem.

   (a) Explain briefly how a FAT filesystem is organised. Pay particular attention to: the identification of free space, the structure of directory entries, and the identification of the sequence of blocks allocated to a file. Illustrate your answer with a diagram showing a possible FAT for the log file of the alarm device. Show how the FAT would be modified if a new block were allocated to the file.

   (10 marks)

   (b) Choose *one* other disk allocation method with which you are familiar and identify its advantages and disadvantages by comparison with the FAT approach.

   (5 marks)

   (c) Why is FAT12/16/32 almost universally adopted as the file system for use with current non-volatile memory cards, such as the SD card?

   (5 marks)

5 Further information

Learning Outcomes assessed in this assessment:

1. Describe the architecture of an operating system (OS) and its services, and evaluate its use in a variety of scenarios.
2. Discuss the process model and the scheduling, IPC and synchronisation services provided by an OS and reason informally about the behaviour of a multitasking system under a variety of scheduling algorithms.
3. Review the principal concepts and methods of memory management and file system implementation.
4. Identify a variety of security threats and examine appropriate OS mechanisms to protect against them.
5. Design, implement and evaluate solutions to problems of I/O device handling, synchronisation, communication and timing for multitasking systems, using appropriate OS services and concurrent programming techniques.

Assessment Criteria/Mark Scheme: The coursework consists of

1. a software development project assessing low-level OS implementation and concurrency (50%)
2. a report on OS theory and concepts (50%)

More detailed marks allocation is provided in the assignment brief.

If you wish, you may collaborate with a partner for the software development and demonstration part of the assignment. If you choose to do so, you must inform the module tutor by email (david.kendall@northumbria.ac.uk) to be received no later than 23.59 on 9th November 2015. This is a hard deadline. If you have not been named as a partner by the deadline, it will be assumed that you are undertaking this part individually. The rest of the assignment, i.e. the software project report and the theory and concepts report, must be entirely your own individual work.

Referencing Style: Harvard

Expected size of the submission: Your report should be about 5 to 7 A4 pages in length (assuming 10pt and normal margins). There is no fixed penalty for exceeding this limit but unnecessary verbosity, irrelevance and 'padding' make it difficult for the marker to identify relevant material and may lead to some loss of marks.

Assignment weighting: 100%

Academic Integrity Statement: You must adhere to the university regulations on academic conduct. Formal inquiry proceedings will be instigated if there is any suspicion of plagiarism or any other form of misconduct in your work. Refer to the University's Assessment Regulations for Northumbria Awards if you are unclear as to the meaning of these terms. The latest copy is available on the University website.

Failure to submit: Note that failure to submit work or submission of work after the required deadline without an authorised late approval will result in a record of incomplete (IC) for the assessment component. Referral in that component will then be required even when the module is passed overall.