JAMES COOK UNIVERSITY AUSTRALIA

Venue _____

Seat Number _____

Student Number |__|__|__|__|__|__|__|__|

Family Name _____

First Name _____

CC2511 Embedded Systems Design

Formal Exams SP2, 2018

Examination

College of Science & Engineering

Examination Duration: 120 minutes

Reading Time: 10 minutes

Exam Conditions:

This is a FORMAL exam.

This is an OPEN BOOK exam - Any books permitted (no restrictions)

Any calculator is permitted.

All notes ARE permitted.

English/Bilingual Dictionary: YES

This exam paper MUST be handed in.

Materials Permitted In The Exam Venue:

Laptops are permitted. Freescale/NXP FRDM development board and USB cable.

Materials To Be Supplied To Students: None

Instructions To Students:

- Answer all questions.
- Your submission is to be a Zip archive file produced by Kinetis Design Studio. Submit by email to mostafa.rahimiazghadi@jcu.edu.au (Townsville) or bronson.philippa@jcu.edu.au (Cairns) and CC'ed to yourself. Do not leave the room until you have confirmed the presence of the email attachment in your inbox.
- It is not permitted to leave the room early.

CC2511 Exam Instructions

- This is an invigilated test. Communication with other persons by any means during this test will be regarded as academic misconduct.
- You must produce a single Kinetis Design Studio project using any mix of C and assembly language.
- The name of your project must be derived from your name in the format Lastname_Firstname_Exam, for example "Rahimi_Mostafa_Exam".
- If your name is too long or contains symbols, you may use initials or abbreviations, but the project name must clearly identify you.
- Do not create a project with a generic name like "Exam" because the examiner will not be able to identify your project in a folder of other submissions.
- Sample program output is shown in monospaced font. Underscores indicate blanks where you are to insert the value indicated.
- It is recommended that you keep appropriate backups during the test, for example by saving your work to an external drive.

Your submission

- Your submission is to be a Zip archive file produced by Kinetis Design Studio, containing a complete project suitable for programming onto a Kinetis development board.
- Your submission is to be emailed as an attachment to mostafa.rahimiazghadi@jcu.edu.au (Townsville) or bronson.philippa@jcu.edu.au (Cairns) and CC'ed to yourself. **Do not leave the room until you have confirmed the presence of the attachment in your own email inbox.**

Follow the steps below when you have finished:

1. Locate your project ("Lastname_Firstname_Exam") in the "Project Explorer" view.
    - If you cannot see the "Project Explorer" view, select it from the Window → Show View menu.
2. Right click on your project ("Lastname_Firstname_Exam") and choose Export.
3. In the Export dialog box, select General → Archive File.
4. In the "To archive file" box, browse to a suitable location (e.g. the Desktop) and give the file a suitable name, e.g. "Lastname_Firstname_Exam".
5. Click Finish.
6. Use your JCU email address to send the resultant Zip file to your lecturer and CC it to yourself.
7. **Do not leave the room until you see your email in your own inbox and you have checked that the attachment is present.**

Assessment procedure

Your project will be imported into Kinetis Design Studio. The code will be read, and your project will be programmed onto the board.

Each question will be separately assessed for the marks indicated. Levels of achievement are indicated below.

High standard	Medium standard	Minimum standard	Fail
The program operates correctly and reliably. It achieves all the objectives listed on the exam sheet. The code is easy to understand and generally well presented.	The program is mostly correct with only minor bugs; OR the program would be correct except for minor programming errors.	The program is mostly correct but several bugs/glitches are present; OR substantial and viable progress has been made towards the task.	The program does not operate correctly and the submitted code does not show viable progress towards the task.

Question 1

Establish serial communications.

(a) Configure serial communications using the OpenSDA USB serial port at **28800 baud**, no parity, no flow control, and a data width of 8 bits. [3 marks]

(b) Program your software to print out the following message once at startup. Insert your own name on the second line as indicated:

```
CC2511 Exam 2018
_____________________
```
(insert your own name)

[2 marks]

Question 2

(a) Configure the analog-to-digital converter (ADC) to measure the voltage produced by the **light sensor** on the FRDM board. You will need to look at the FRDM schematic to identify the pin that is connected to the light sensor. [5 marks]

(b) Detect when the "**m**" key is received over the serial port. Once this key has been received, print out a message "Monitoring mode". In monitoring mode, run the following steps **every 200 milliseconds**. [5 marks]

1. Perform an analog to digital conversion and store the resultant value in a variable $X_{measured}$. [10 marks]
2. Apply the equation below to calculate a value called X_{smooth}:

$$X_{smooth} = \alpha X_{smooth} + (1 - \alpha) X_{measured},$$

where $\alpha = 0.9$ defines the amount of smoothing to apply and $X_{measured}$ is the latest ADC measurement. You should initialise X_{smooth} to zero at the beginning of monitoring mode. [10 marks]

3. Print the values of $X_{measured}$ and X_{smooth} over the serial port. [5 marks]
4. Control the **blue LED** according to the following relationship: [5 marks]

$X_{measured} < X_{smooth}$	Blue LED off
$X_{measured} \geq X_{smooth}$	Blue LED on

(c) The user can exit monitoring mode by pressing any other key. Set up your program to detect a key press and then leave monitoring mode, e.g. stop following the steps in part (b) every 200 ms. [5 marks]

Hint: You can use the FreeCntr32 component to measure time, and/or the TimerInt component to generate periodic interrupts.

Question 3

Extend your program to implement a math game for kids. The game is to practice mental subtraction. You will pose a subtraction problem, for example

$$15 - 9$$

and the user would have to type the number 6.

You will implement a simple version of this game where answers are always single digits. This means you do not need to establish a string buffer to read input; you can simply receive a single character and check if it is the correct answer.

The rules of the game are as follows:

a) The game begins when the "**g**" character is received over the serial port. Once this character is received print out a message "Entering game mode". [5 marks]

b) Randomly generate two integers a and b such that $a > b$ and $a - b < 10$. This is necessary so that the answer of the subtraction is a single digit. (See hint on the following page.) [10 marks]

c) Display the subtraction problem $a - b$ to the user and await their response. If the user types in the correct answer, print "Correct". If they made a mistake, print a message "Incorrect". [5 marks]

d) Maintain a score, where the user gains a point for a correct answer and loses a point for an incorrect answer. Display the score to the user after each answer. Once the score reaches +5 or -5, the game ends. [5 marks]

For full marks, implement the following features:

- Visually indicate the user's progress towards a +5 or -5 score by PWM on the green and red LEDs. Specifically, have the LEDs off when the score is zero. Progressively increase the brightness of the green LED as the score increases to +5, and similarly on the red LED as the score drops to -5. Maximum brightness is achieved for scores of +5 or -5 respectively. [5 marks]

- Implement a time out feature, where the user has a time limit in which to type their response. They lose a point if they fail to respond in time and a new question is asked. [10 marks]

- Once the user reaches a score of +5, increase the difficulty of the problem by adding a new term, such as in the problem

$$11 + 8 - 17.$$

The user now must reach a score of +10 to finish the game. [10 marks]

Hints

The function rand() returns random numbers. The function prototypes are:

```
#include <stdlib.h>

int rand(void);
void srand(unsigned int seed);
```

Use these functions as follows:

```
// So that the random number engine does not produce the
// same sequence every time the board starts up, you should
// seed the generator with a number derived from something
// unpredictable, e.g. the timing of user input.
srand(unpredictable_seed);


// To generate a random number in the range 0 .. 9 inclusive:
int r = rand() % 10;
```

END OF EXAMINATION