Electrical, Electronic & Computer Engineering
School of Engineering & Physical Sciences

HERIOT WATT UNIVERSITY

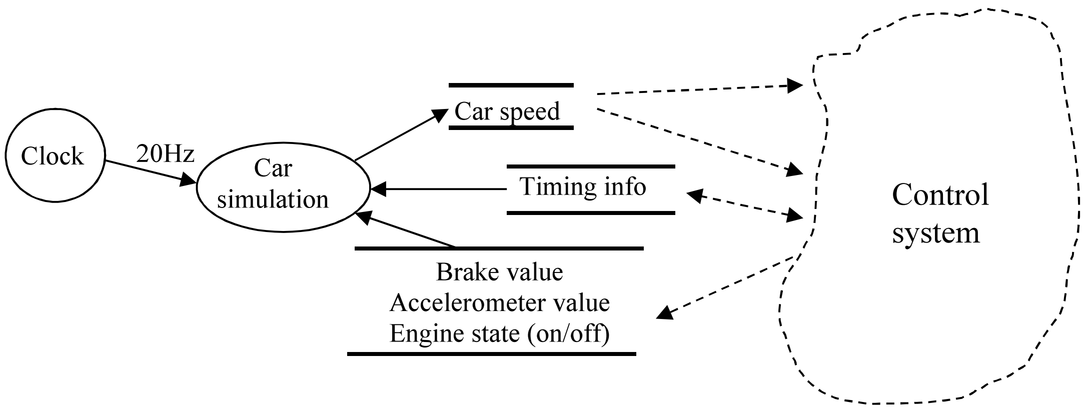
B31DG2 : Embedded Software
Assignment 3.

1. Problem

Modern cars use many microcontrollers in their systems. This assignment will involve the construction of a very simplistic controller for some aspects of car operation. You will use the Real-Time Operating System (RTOS) on the MBED to implement the software.

2. Simulation of the car

Clearly it will not be possible to provide a real car for testing. To overcome this problem you will implement an additional task that will provide a simulation of a car. This is a very powerful benefit of using a multi-task approach to the development of real-time systems.



You will design a VERY simplified model of the car to allow the control system to operate. For example, accelerometer causes speed to increase and brake make speed decrease. **This task must run at 20Hz.**

3. Controller Specification

Build a controller for a part of a simple car system. The system will use the MBED RTOS system as the basis of the software. The system should provide the following features ::

	Feature	Repetition rate (Hz)
1.	Read brake and accelerator values from variable resistors • Ensure that A/D inputs are ALWAYS less than 3.3v.	10
2.	Read engine on/off switch and show current state on an LED.	2
3.	Filter speed with averaging filter • average of last ‘n’ readings (e.g. n =3) (raw speed value will be computed by the “car simulator” process)	5
4.	Show the following value with a RC servo motor • Average speed (Use servo output from MBED motherboard)	1
5.	Monitor speed and if it goes over 70 mph switch on an LED on the RedBox unit.	0.5
6.	Display on MBED text display • odometer value • average speed in MBED text display	2
7.	Send speed, accelerometer and brake values to a 100 element MAIL queue • A MAIL queue is a structure available in the MBED RTOS	0.2
8.	Dump contents of feature_7 MAIL queue to the serial connection to the PC. (Data will be passed through the MBED USB connection). See later technical note.	0.05
9.	Read a single side light switch and set side lights accordingly.	1
10.	Read the two turn indicator switches and flash appropriate indicator LEDs at a rate of 1Hz. If both switches are switched on then flash both indicator LEDs at a rate of 2Hz (hazard mode).	0.5

List of Input/Output signals

Signal	Direction	Purpose
A/D 1	In	Accelerator pedal
A/D 2	In	Brake pedal
Dig 1	In	Engine on/off switch
Dig 2	In	Side light on/off switch
Dig 3	In	Left indicator switch
Dig 4	In	Right indicator switch
Servo	Out	Average speed
MBED LED 1	Out	Side light indicator
MBED LED 2	Out	Left turn indicator
MBED LED 3	Out	Right turn indicator
MBED LED 4	Out	Engine on/off indicator
RedBox LED	Out	Overspeed indicator

3. Task 1 : Design and build

Write a C/C++ multi-task program to implement the above problem. Show it running on the MBED system. The structure MUST use the MBED RTOS software. **Test early and test often**.

***** All programming activity MUST be logged into the MBED version Control facility *****

4. Task 2 : Test

The test sessions will involve

1. Run the code and show the features.
2. Show the logged data coming up through the serial link. (Not part of requirement, but you could collect this data and display it in Excel spreadsheet.)
3. Show the revision graph generated in the MBED integrated version control facility.

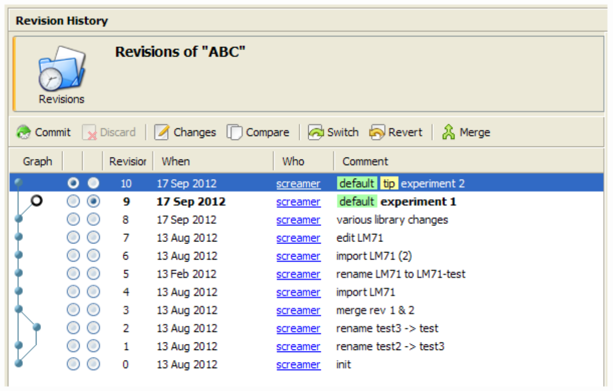
5. Task 3 : Document

Submit a short report when your software is being tested. Include the following

1. A fully documented listing of your program. Code should have a minimum point size of 10. Please include a note about which method you have used to implement repetitive tasks. (see technical note 5)

2. A print-out of the graph showing the 'MBED Revision History' (1 page). You should plan to commit a revision (with a comment) at the end of each development session. Will be in this form.

   I expect this graph to cover 2 weeks from the first entry to the final entry.



Revisions of "ABC"

Graph			Revisior	When	Who	Comment
	●	○	10	17 Sep 2012	screamer	default tip experiment 2
	○	●	**9**	**17 Sep 2012**	screamer	default **experiment 1**
	○	○	8	17 Sep 2012	screamer	various library changes
	○	○	7	13 Aug 2012	screamer	edit LM71
	○	○	6	13 Aug 2012	screamer	import LM71 (2)
	○	○	5	13 Feb 2012	screamer	rename LM71 to LM71-test
	○	○	4	13 Aug 2012	screamer	import LM71
	○	○	3	13 Aug 2012	screamer	merge rev 1 & 2
	○	○	2	13 Aug 2012	screamer	rename test3 -> test
	○	○	1	13 Aug 2012	screamer	rename test2 -> test3
	○	○	0	13 Aug 2012	screamer	init

3. A note describing your "car model" (1 page)

4. A dataflow diagram of your system. (1 page)

5. Send me an email with the following
   - Subject line of the form **B31DG : assignment 3 : your name**
   - To include a single attachment of a TEXT file which will be a copy of all your source code. Text file name should exactly follow this format

     ::: **Herd_Jim_B31DG_assignment3.txt**

     with your name in place of mine.

6. Technical notes

1. **All digital and analogue inputs MUST be 3.3volts or less. Larger values can damage the MBED unit.**
2. MBED RTOS information at
   - https://developer.mbed.org/handbook/RTOS
3. The MBED RTOS is based on the Keil RTX system. The following link deals with the RTX system but is equally applicable to the MBED RTOS
   - http://arm-software.github.io/CMSIS_5/RTOS/html/group__CMSIS__RTOS.html
4. Read the RTOS documents first, and build the solution task by task.
5. There are a number of ways to implement repetitive task in MBED RTOS.
   1) You can use RTOS TIMERS to implement the tasking, but there are two problems.
      - They cannot be given a priority
      - There is an internal restriction on the number of timer functions allowed. However, this can be changed.
   2) You can use RTOS TASKS with an appropriate delay. To use this method you need to measure the time it takes to execute the task code and then delay the remaining time until the next execute time. Not ideal, but allows task priority.
   3) You could use the method outlined in the lecture notes that constructs an additional scheduler. A very high priority task would run this little scheduler.
   4) I would prefer you use either method 2 or 3.
6. You may use any of the LEDs on either the MBED or on a "red-box" unit.
7. Use the switches and potentiometers on the lab "Red box" units.
8. Keep the MBED display layout as simple as possible as it is quite slow.
   1) Modify if display causes problems.
9. Run the serial comms port as fast as possible (try 115200 baud)
10. To use the serial port (through the USB connection) you may need to install a serial driver. Details for Windows machines are here –
    1) http://developer.mbed.org/handbook/Windows-serial-configuration
    2) A number of Windows terminal emulator programs are available that can interact with the serial port e.g. PUTTY, Terminal, etc
    3) The computers in EM1.81 will have the driver installed.

7. Organisation notes

11. Work as an individual.
    1) **Students MUST work on this project on their own. Please do not use code from another student or offer code to another student. Code may be run through a similarity checker.**
12. **Testing will be on Tuesday 28th March and Wednesday 29th March.**
    1) **Assignment is worth 20% of final mark**
       - **Lab demonstration 15%**
       - **Report 5%**
    2) **Lab demonstration mark will be set to zero if either no report is submitted or the submitted report is unsatisfactory.**