EEE20003 Experiment 4 - Assessment

Clock Generator

Group Number	Demo	Report
Student 1: Name : ID #	/10	/10
Student 2: Name : ID #	/10	
Lab Session (Day and time e.g. Monday 14:30) **(Note: Failure to fill this in correctly will result in the circular file being fed)**		

Demonstration & Oral assessment

The assessment is divided into the following sections:		Demo	Report
Section 2 – API ○ Provides a convenient API for the GPIO function with the correct functionality (direction, reading and writing pins) ○ Students display a good understanding of the I2C operation and can describe how registers in the GPIO can be accessed and how these registers control pin functions. ○ Students demonstrate a good understanding of the advantages and limitations of using a I2C based GPIO expander relative to microcontroller ports.	/4		
Section 3 – Animation ○ The animation covers the full range of the bar graph in response to sensible range of movement of the board (accelerometer). This may be demonstrated in section 4.	/2		
Section 4 – Game ○ The game operation is as described int the lab sheet: ▪ Random walk affect position ▪ Tilt affects position ▪ Game difficulty is reasonable (at least initially) ▪ Final score is displayed when game over ○ PIT call-back is used to implement the game logic (1 mark) ▪ Speed advances ▪ Mainline reports progress	/4		
(Note: The total weighting is 10% of the unit)	Total	/10	/10
Lab assessment will involve answering questions about the demonstration and the understanding of the program. It is expected that each group member will be able to respond to questions on any part of the submitted work.			

Written Report

Section

The written report covers the same areas as outlined for the oral assessment.
You cannot receive more marks for the report than allocated to the achieved sections. For example, if only section 2 is attempted then the report is marked out of 4.

General requirements of the final program

- The program should be constructed in modules with a sensible division of responsibility. See examples covered in tutorial and the modules outlined in the experiment sheet.
- The module header should only publish information that must be shared e.g. implementation details such as register addresses should not appear in headers.
- The interface to each module should be clearly described by the associated header file.
- Global variables should be avoided where possible. Obviously, ***module-global*** variables (**static**) are needed for interrupt functions and related data.
- Program-level global variables ***must not be used***.
- The program makes use of PIT interrupts as outlined in the laboratory sheet (1 mark).
- Doxygen style comments should be used to document the code.

Report Requirements

- **MUST USE THE COVER SHEET PROVIDED**
- A brief description of the *overall program operation*. This would include how the various modules interact.
- An overall diagram indicating the structure of the program and the responsibilities of the various modules of the program and how they are related. This diagram should show the interface functions provided by each module.
- A description of each module in the program briefly describing their responsibilities and describing the interface to the module.
- A doxygen generated page should be provided for the GPIO API
- It is not necessary to submit material related to the provided USBDM library e.g. LCD etc. This may be treated as a well-known library.
- Documentation of testing procedure is required with supporting evidence e.g. a table of test cases and results.
- Confirmation of expected operation should be obtained using PulseView and example I2C waveforms provided.
- Program Listings. These **should use a fixed width font e.g. Courier or Consolas, and lines should not wrap.** It may be necessary to format listing in landscape.
  - The program should be **adequately commented**.
  - The program should be **correctly indented**.
  - Commenting of instruction lines should describe their purpose - not just repeat the instructions in English.
  - The program files should have a suitable banner with a brief description of the purpose of the module.
  - Each routine should have a banner indicating entry & exit conditions and purpose.
  - Constants & variables should be documented.
  - Meaningful names should be used for variables, constants and functions.