ENGT5259 EMBEDDED SYSTEMS PROJECT 2021-2022

Author: J A Gow *v1.3*

Title:	***Firmware design for a DC motor speed controller***
Style:	***Individual project.***
Assessed by:	***Firmware code and viva/demonstration***
To be submitted by:	*Code to be submitted online by:* ***13/5/2022*** *Viva/demonstration to take place:* *week beginning* ***16/5/2022***
Feedback available:	***At regular intervals throughout the year and in any scheduled laboratory session. Demonstration/viva feedback will be immediate and verbal.***
Marks available by:	***June 2022 once all demonstrations are complete.***
Anonymously marked:	***No (individual project)***
Issued by:	***Dr J A Gow (jgow@dmu.ac.uk x7085)***
Percentage of module:	**100%**

Contents

Version record

Version	Notes	Date	By
1.0	Initial version	21/01/22	JAG

1.1	Typos corrected	23/01/22	JAG/MAO
1.2	Added submission deadlines	28/01/22	JAG

PLEASE READ THIS CAREFULLY

The coursework for this module is a single project, **'Firmware design for a DC motor speed controller'**. This is an individual project, with a short demonstration of the final firmware you will produce. There are NO formal reports required. However, you will be encouraged to keep and maintain an online journal (a.k.a. a logbook) on the Blackboard shell to which you will be expected to make ***regular*** updates. This logbook is not assessed in itself, but is there to enable you to record your progress and thought processes as you progress through the project, and will help you to evidence your work during the viva/demonstration.

1 Aims and objectives

You are to design the firmware for a speed controller for a DC motor**.** This project is firmware only – note all the hardware you require will be provided and you will not be expected to design or build any hardware.

1.1 Platform

The hardware platform you will use for your design is the DMU-designed ATMega328P Microcontroller Development Board. These are available in Queens Q2.01 and Q1.01. This board is based around an ATMega328P microcontroller, and this Atmel AVR-based microcontroller is the target platform for your firmware. The board contains a DC motor attached to a small fan, and an infra-red (IR) sensor that will allow you to count the number of times the IR beam is broken by each of the three fan blades. The board also contains:

- a 3x4 keyboard matrix
- a rotary encoder switch
- a 7-segment LED display
- an accelerometer
- a number of individual coloured LEDs
- a 2 line by 16 character LCD display

All of the above items are interfaced to, and therefore can be controlled by, the ATMega328P microcontroller

Not all of these features may be required in order to implement this specification.

1.2 Schematic

A schematic for the development board is available and may be downloaded from the Blackboard shell. This should be consulted, along with the datasheets for the associated peripheral ICs, in order to determine how the firmware should be constructed to manipulate the required hardware.

1.3 Kernel

A minimal low-footprint 'operating system' is provided. This provides memory management, message queues and a round-robin task manager: common services provided by most embedded operating systems. It is there to help you, and the source code is provided. This should be made use of in your final design. Source code to the kernel is available and can be downloaded from Blackboard, along with a template project to get you started.

1.4 Features and specification

Your firmware must implement the following features:

1 Display

1.1 The 2x16 LCD display should display two three-digit values, along with appropriate text, to indicate the demanded speed (what is set by the operator via the keypad or the rotary switch) and the actual speed (as measured). The manner and position on the display of these two values and accompanying text is up to the designer, but at all times when the system is powered, the display should be legible and there should be no ambiguity between the values as displayed.

1.2 Leading zeroes should **not** be suppressed.

1.3 When the operator changes the speed via the keypad, a cursor on the display over the demanded speed should allow the operator to visualize the entry of one digit at a time. This will include the backspace function (see 2.6). If the operator changes the speed via the rotary encoder, the demanded speed should change directly without a cursor.

1.4 If the operator enters a speed outside of the range specified in section 8 , the display should display the characters ‘ERR’ in place of the demanded speed for 2 seconds +/- 0.5 second. The display should then revert to the value displayed before the out-of-range speed was entered.

2 Keypad:

2.1 All keyboard switching should be fully debounced.

2.2 A key is accepted as ‘pressed’ at the point when the debounce is complete if it is determined that the transitions on the keyboard matrix are consistent with an actual keypress.

2.3 If more than one key is pressed at the same time, the first key to be detected as pressed is the one to be accepted.

2.4 The desired speed should be set by the operator entering the speed via the keypad.

2.5 The '#' key is an 'enter' key – the speed entered by the operator will be accepted by the firmware once this key is pressed.

2.6 The '*' key is a backspace key – this will delete the previous digit entered. If no digits are entered, this key will do nothing.

3 Rotary encoder

3.1 The rotary encoder can also be used to set the speed. This should be arranged so that one click clockwise increases the speed by 1rpm, one speed anticlockwise decreases the click by 1rpm.

4 7-segment LED display

4.1 The 7-segment LED display should display the current key being pressed, for the duration for which it is pressed. '#' should be displayed as 'E', while '*' should be displayed as 'b'.

4.2 When no key is pressed, the decimal point should be illuminated.

4.3 The decimal point must be extinguished for the duration for which a key is pressed.

5 Analog pot

5.1 This is not used in this specification.

6 Push switches S1 and S2

6.1 These are not used in this specification but may be used for any additional purpose as desired by the designer (e.g. debugging).

7 Individual LEDs

7.1 Individual LEDs are not required for this specification, and thus may be used for any purpose as desired by the designer (e.g. debugging).

8 Control

8.1 On power up/reset:

8.1.1 The message "Starting…" should be displayed on the screen until the firmware is fully initialized. This period should not exceed 2 seconds.

8.1.2 Once the firmware is fully initialized, the display should read in accordance with section 1 of this list.

8.1.3 the motor should be stationary (not rotating).

8.1.4 the 2x16 display should read zero for both demanded and actual speed.

8.1.5 The 7 segment display should be blank, with the decimal point illuminated.

8.2 Motor speed control

8.2.1 Motor speed control should be achieved only by closed loop digital control. Open loop control is not permitted.

8.2.2 Maximum displayed speed error (displayed demanded value to actual value) is 5%

8.2.3 Maximum actual speed error (as measured by an external tachometer) is 5%

8.2.4 Minimum motor speed is 20rpm.

8.2.5 Maximum motor speed is given by the maximum speed possible when the motor is driven from a PWM source with a duty cycle of 1. This will have to be determined experimentally.

8.2.6 Operators shall not be able to enter a speed out of this range without the 'ERR' message being displayed (section 1.4) and the value being rejected.

9 Firmware restrictions

9.1 The firmware shall be constructed in such a manner as to facilitate ease of expansion with additional I/O.

9.2 The provided kernel must be used.

9.3 Languages:

9.3.1 C or C++ should be used. Some functions may be written in machine code using AVR assembler language where appropriate.

9.4 **Libraries**:

9.4.1 Arduino library functions may NOT be used, with two exceptions. These are (1) members of the 'Serial' class may be used for debugging only providing that any calls to these functions are removed or commented out in the final design, and (2) the LiquidCrystal_I2C library may be used in connection with the implementation of section 1 of this specification.

2 UKSPEC learning outcomes

SM1fl:	A comprehensive understanding of the relevant scientific principles of the specialisation.
SM2fl:	A critical awareness of current problems and/or new insights most of which is at, or informed by, the forefront of the specialisation.
EA1fl:	Ability both to apply appropriate engineering analysis methods for solving complex problems in engineering and to assess their limitations.
EA2fl:	Ability to use fundamental knowledge to investigate new and emerging technologies.
EA3fl:	Ability to collect and analyse research data and to use appropriate engineering analysis tools in tackling unfamiliar problems, such as those with uncertain or incomplete data or specifications, by the appropriate innovation, use or adaptation of engineering analytical methods.

D1fl:	Knowledge, understanding and skills to work with information that may be incomplete or uncertain, quantify the effect of this on the design and, where appropriate, use theory or experimental research to mitigate
D2fl:	Knowledge and comprehensive understanding of design processes and methodologies and the ability to apply and adapt them in unfamiliar situations.
D3fl:	Ability to generate an innovative design for products, systems, components or processes to fulfil new needs.

3 Submission requirements:

There are two parts to this submission, which together comprise a single component. The first is the firmware only, and the second is a viva/demonstration during which it will be expected that you will be able to answer technical questions relating to the code you have written. Hardware will be available during the demonstration if you wish to refer to this during the viva/demonstration. Failure to submit either will result in failure of the module.

Note: no written report is required. All assessment is determined from your code and your performance in the viva.

3.1 Firmware submission

A portal will be opened on Blackboard closer to the date specified in section 4 . The firmware should be zipped up and a single .zip file submitted. Note the following:

- Please ensure you have **checked that the .zip file contains all the files** (and there will be more than one) of your firmware project. You do not need to include the provided kernel in your .zip file (unless you have modified it).
- Please ensure that you can:
    - **unpack the .zip file**
    - **that the firmware compiles** once you have unpacked it (i.e. the newly unpacked project will compile, not your original one).

There will be no leniency given if the firmware you have submitted fails to compile. You will be expected to have checked it.

3.2 Viva/demonstration

The vivas/demonstration will be held the week after the module ends. As numbers of students may dictate that this takes place on a day different to your scheduled sessions, we will communicate the exact day and time to all students registered via the Blackboard shell closer to the time.

The viva is the main point of assessment. Failure to attend the viva will lead to failure of the project as a whole. The vivas are intended to probe each of the learning outcomes with respect to the project you have submitted, as well as to determine authorship. If you have worked consistently and submitted your own work, you will have no problems with the viva.

4 Schedule of submission dates/deadlines:

Week/date	**Description**
Friday 13th May	Submission of firmware code via Blackboard
Week 33	Demonstration/vivas (exact date and time to be communicated via Blackboard)

5 The lab equipment:

5.1 DMU labs

We have a standard laboratory (Q2.01), with oscilloscopes, signal generators and power supplies. Development boards are available in Q2.01. There should also be boards in Q1.01.

6 Marking scheme:

Criterion:	Clear Fail <40%	Marginal Fail 40 - 49%	Pass to Merit 50 - 59%	Merit 60 - 69%	Distinction >70%
Style of provided source code	Shambolic, or very little original source code presented. No clearly logical execution path and code may not compile. No, or very few comments. No encapsulation, poor or no effective use of operating services.	Code is very unclear and can not be easily read to determine whether or not the code is capable of functioning as claimed (if errors were corrected). Very few or no comments. Indentation style makes code difficult to read. No encapsulation, poor use of operating system services	Code compiles, is reasonably clear in most places. Indentation style is generally acceptable in most places. There are enough comments to allow the basic principle of the design to be seen. Some attempt has been made at encapsulation in most places. Attention has been paid to where efficiency is required.	Good, readable code with a clear indentation style. There may be some convoluted areas but would in general be acceptable in industry. Encapsulation and use of operating system services are generally well applied. Coding style and choice of structure reflects requirements for efficiency	Excellent quality code, instantly readable with strong use of encapsulation and operating system services. Very few errors or unnecessarily convoluted structures. Good use of language features and choice of design methodology matches the requirements of efficiency.
Specification requirement (as listed in marking scheme)	Lack of or very limited advance on code given in the examples. No understanding of the code submitted (if any) and/or completely non-working when tested on hardware. Structureless and poorly designed. Could not be easily fixed to make it work.	An attempt has been made to implement the requirement, but this is deficient in too many ways to enable the design as it stands to be easily fixed to work as desired.	Specified requirement has been implemented and can be shown to work as per the specification with few or no compilation errors that were not easily fixed by the student at the viva. There may be a number of design or stylistic errors, but learning outcomes have been hit and the student clearly understands the code they have written, its strengths, and how it is deficient when compared with the specification	Specified requirement has been fully implemented and works well with no compilation or significant design errors. Meets the specification. Learning outcomes have been hit and the student clearly and fully understands the code they have written and how it was designed. There may be one or two minor errors, easily fixed and understood	Specified requirement fully implemented and completely meets the specification. Fully understood at the viva. Any errors are minor and do not significantly detract from the implementation of the specification.
Performance at viva with respect to Q&A	Does not understand the code they have submitted. Technical knowledge is substandard. Very little in the way of technical knowledge	Some technical knowledge and an attempt to answer wider questions in the subject area. There is some understanding of the code that has been submitted but this is weak.	Student performance is of an acceptable standard, it is clear that the student understands their own submission and that the learning outcomes are hit, but the wider knowledge may be weak. In general the assessors would feel confident if this student was employed in industry in a role involving embedded system design	Fully understands the code in their submission and can articulate it well. There may be some gaps in the wider knowledge of embedded systems, but in general their knowledge is sufficient for them to be able to develop a design independently if they had access to written materials	Very good wider knowledge, fully understands the code and design they have submitted and can articulate this fully. Would be able to easily develop an embedded system independently and would know where to find further information to broaden their knowledge.

7 Late submission of coursework policy

Late submissions will be processed in accordance with current University regulations which state:

“the time period during which a student may submit a piece of work late without authorisation and have the work capped at 40% [50% at PG level] if passed is ***14 calendar days****. Work submitted unauthorised more than 14 calendar days after the original submission date will receive a mark of 0%. These regulations apply to a student’s first attempt at coursework. Work submitted late without authorisation which constitutes reassessment of a previously failed piece of coursework will always receive a mark of 0%.”*

The viva/demonstration will take place on a fixed date to be notified. If you fail to attend your allotted demonstration time, you will be able to reschedule within 14 calendar days in accordance with the above regulation

8 Academic Offences and Bad Academic Practices:

These include plagiarism, cheating, collusion, copying work and reuse of your own work, poor referencing or the passing off of somebody else's ideas as your own. If you are in any doubt about what constitutes an academic offence or bad academic practice you must check with your tutor. Further information and details of how DSU can support you, if needed, is available at:

http://www.dmu.ac.uk/dmu-students/the-student-gateway/academic-support-office/academic-offences.aspx and

http://www.dmu.ac.uk/dmu-students/the-student-gateway/academic-support-office/bad-academic-practice.aspx

ENGT5259 – Embedded Systems
Marking guide (to be completed after viva)

ID: _________ Name:_______________________________ Date:_______

Learning outcomes met overall: ☐ Yes ☐ Partially ☐ No

Note: If the submitted source code can not be made to compile at all, and the student is unable to remedy the situation at the viva, then marks can only be awarded for sections 1 and 3 below. Section 2 marks should be recorded as zero.

	Criterion			***Weight (%)***	***First Marker mark (%)***	***Weighted mark (%)***	***Moderator mark (%)***	***Moderator weighted mark (%)***
1	Style of provided source code			10%		0.000%		0.000%
2	Requirements (as numbered in specification)	1	Display	10%		0.000%		0.000%
		2	Keypad	10%		0.000%		0.000%
		3	Rotary encoder	10%		0.000%		0.000%
		4	7 segment display	10%		0.000%		0.000%
		8.1	Power up/reset	10%		0.000%		0.000%
		8.2	Motor speed control	20%		0.000%		0.000%
		9	Compliance with firmware restrictions	10%		0.000%		0.000%
3	Performance at viva with respect to Q&A			10%		0.000%		0.000%
	Totals					**0%**		**0%**

First marker: ____________________ Moderator:__________________________

Agreed mark ___________________

Further comments: