University of Technology, Sydney

Faculty of Engineering and Information Technology

Subject:	**48434 Embedded Software**
Assessment Number:	**5**
Assessment Title:	**Lab 5 – Human-Machine Interface**
Tutorial Group:	

Students Name(s) and Number(s)

Student Number	Family Name	First Name

Declaration of Originality:

The work contained in this assignment, other than that specifically attributed to another source, is that of the author(s). It is recognised that, should this declaration be found to be false, disciplinary action could be taken and the assignments of all students involved will be given zero marks. In the statement below, I have indicated the extent to which I have collaborated with other students, whom I have named.

Statement of Collaboration:

Signature(s)

Marks

Use of Tortoise SVN	/2
Push buttons HAL	/1
LCD HAL	/2
Menu system	/3
TOTAL	/8

Office use only ☺

key

Assessment Submission Receipt

Assessment Title:	**Lab 5 – Human-Machine Interface**	**Mark**
Student's Name:		
Date Submitted:		
Tutor Signature:		Office use only ☺

Lab 5 – Human-Machine Interface

Pushbuttons. Debouncing. Liquid Crystal Displays. Hardware Abstraction Layers.

Introduction

You are to design a hardware abstraction layer (HAL) for the ModCon's human-machine interface that consists of pushbuttons and a graphical liquid crystal display (LCD). A menu system will be developed for the LCD that allows a user to scroll through various items, and select values. The pushbuttons will be debounced in software to keep the hardware simple.

Objectives

1. To implement a hardware abstraction layer for a graphical LCD.
2. To implement software switch debouncing.
3. To implement a menu system as a human-machine interface.

Equipment

- 1 ModCon microcontroller board – UTS
- 1 ModCon analog interface board – UTS
- 1 ModCon human-machine interface board – UTS
- 1 HCS12 USB programmer / debugger – PEMicro
- 2 USB cables
- 1 5V DC power supply, 1 12V DC power supply, PC connector
- Freescale CodeWarrior IDE for the MC9S12

Safety

This is a Category A laboratory experiment. Please adhere to the Category A safety guidelines (issued separately).

Cat. A lab

L5.2

Pushbuttons

There are five pushbuttons, to be used for the following functions:

Button Name	Function
Set	Accesses the menu system to choose parameters, and returns from the menu system
Data	Cycles through a display of relevant data, and returns from the menu system
Up	Move up a menu item
Down	Move down a menu item
Select	Selects a menu item, or selects an item value

Even though the pushbuttons are interfaced with hardware debouncing, it will be necessary (for this lab) to implement software debouncing (consider the current PCB as a proof-of-concept prototype, with later “production PCBs” going for the cheaper option of no hardware debouncing).

Liquid Crystal Display

The liquid crystal display is a graphical LCD. You are using a CV12864C from Clover. You will need to find datasheets for this LCD via the web.

The data lines of the LCD are connected to the MC9S12's Port A. The control lines of the LCD are connected to Port B.

The contrast of the LCD is able to be controlled digitally – i.e. set by software running on the MC9S12. Port B is used to interface to the digital contrast circuitry.

The graphical LCD has LED backlighting. The backlighting is able to be turned on and off digitally – i.e. by software running on the MC9S12. Port B is used to interface to the LED backlighting control circuitry.

Human-Machine Interface

A broad set of specifications will be given for the HMI rather than a precise prescription.

The HMI must:

- have a default display which shows the ModCon status – its number, software version, debug mode and communication mode.
- implement a menu system to enable the setting and display of the following ModCon parameters – number, debug mode and communications mode.
- only update all settings at once, if the user desires a setting change. The user must be allowed to either accept or discard all changes made to parameters.
- if the user has not pushed a button for an interval of 10 seconds, then the display must return to the default display.
- control backlighting to the LCD and provide digitally adjustable contrast.

Schematics for the HMI can be found on the Embedded Software web site.

L5.4

Hardware Abstraction Layers

Liquid Crystal Display

The following public functions, which form the basis of a hardware abstraction layer, have been provided to you:

Name	Service
`LCD_BackLight()`	Controls the backlight of the LCD.
`LCD_Clear()`	Clears the LCD.
`LCD_OutChar()`	Writes a character to the LCD.
`LCD_OutString()`	Writes a string to the LCD.
`LCD_SetContrast()`	Sets the contrast of the LCD.
`LCD_Setup()`	Sets up the LCD.

You should expand the hardware abstraction layer for the LCD to suit your application and to provide a more general interface. The following is a list of possible functions you may wish to write:

Name	Service
`LCD_CursorHome()`	Homes the cursor.
`LCD_GoTextXY()`	Positions the cursor at coordinates corresponding to the text.
`LCD_Printf()`	Writes a standard printf string to the LCD.
`LCD_SetFunction()`	Sets a function of the LCD, such as cursor size, cursor blink, text attribute.
`LCD_SelectText()`	Selects text on the LCD by reverse highlighting it.
`LCD_SetLine()`	Sets the current line of the LCD.
`LCD_Shift()`	Sets the shift mode of the LCD, or shifts the cursor or display.

The following private functions, which are part of the hardware abstraction layer implementation, have been provided to you:

Name	Service
`SendCommand()`	Sends a command to the LCD.
`StatusCheck()`	Checks the status of the LCD.
`WriteAuto()`	Writes data to the LCD in auto mode.
`WriteByte()`	Writes a bytes to the LCD.

You should expand the hardware abstraction layer implementation to support the public functions. The following is a list of possible functions you may wish to write for internal use by the LCD module:

Name	Service
`SetAddressPointer()`	Sets the start address for writing to)or reading from) external RAM.
`SetCursorLines()`	Sets the number of lines used by the cursor.
`SetCursorPosition()`	Set the cursor position.
`SetTextArea()`	Used to adjust the columns of the text display (up to the limit defined by H/W).
`SetTextHomeAddress()`	Sets the starting address in the external display RAM for text display. The text home address indicates the leftmost and uppermost position.
`SetTextMode()`	Sets the mode of the text.

There may be other internal functions that are needed – it depends on your implementation.

L5.6

Pushbuttons

You should write a hardware abstraction layer for the pushbuttons which uses the following functions:

Name	Service
`Buttons_Get()`	Gets the state of the pushbuttons.

You may be required to write other functions for internal use by the Buttons module.

Software Requirements

1. The software is to incorporate all the features of Lab 4.
2. No header files for the menu system have been specified.
3. The software must have the functionality given by the general description.
4. TortoiseSVN must be used for version control.

Marking

The software should be ready for marking on the date specified in the Timetable in the Learning Guide.

Software marking will be carried out in the laboratory, in the format of an oral exam.

Marking criteria are on the front page. Software in this lab will be marked from an architectural point-of-view, rather than a functional point-of-view, i.e. modules are structured to be independent, etc.

For full marks for the TortoiseSVN component of the assessment, a history of development from Lab 1 through to Lab 5 must be shown, with appropriate and relevant comments that document the code changes.

Also refer to the document “Software Style Guide”.