Real-Time and Embedded Systems Design Project Document

Subject: CSE211 (Real-Time and Embedded Systems Design)

18P2491 Mazen Mostafa Fawzy Moussa

18P7232 Mohamed Mokhtar Abdelrasoul

18P7713 Mohamed Yasser AbdelSamad

18P6713 Youssef Maher Nader Halim

18P5886 Andrew Saied Labib Barsom

Group: 2

Section: 2

Due Date: 4/6/2022

Introduction

This project is a Temperature sensor made using Tivaware and using FreeRTOS. The temperature sensor value is read through the Tiva's built-in ADC. The value is then sent to the connected LCD after being encoded the correct way. Multiple synchronization and data-safety patterns are being used from the freeRTOS layer (e.g. semaphores and queues) to have maximum reliability.

Assumptions

- To simulate the heater being turned off we use PortF's red LED. (i.e. above set point value and heater turns off)
- To simulate the heater being turned on we use PortF's blue and yellow LEDs. Yellow when at the exact setpoint and blue when below the set point.
- To simulate the buzzer buzzing we send a string to PuTTy through UART.

File and Functions Documentation

“Init.c” and “Init.h”

“Init.h” is the header file of the “Init.c” file and contains the main function declarations and includes.

The “Inits.c” file contains initializations for the UART peripheral, ADC peripheral, ports, pins, and switch interrupts used within the project.

The initialization for PortF, PortA and PortE are in PortFInit, PortAInit and PortEInit respectively. Each function initializes the needed pins in its respective port with the correct configuration. Pins initialized in PortF are:

- Pin 0 [Input]
- Pin 1 [Output]
- Pin 2 [Output]
- Pin 3 [Output]
- Pin 4 [Input]

Pins initialized in PortA are:

- Pin 0 [UART0 RX]
- Pin 1 [UART0 TX]

Pins initialized in PortE are:

- Pin 3 [ADC]

There are another three peripheral initialization functions. Namely:

- ADCInit: Initializes the temperature sensor (calls PortEInit)
- UART0Init: Initializes the UART communication using UART0 through Tivaware
- SwitchInterruptInit: Initializes the built-in switch

“LCD.c” and “LCD.h”

The “LCD.h” file includes several header files. Namely:

- stdint.h
- stdbool.h
- string.h
- inc/hw_types.h
- inc/hw_memmap.h
- driverlib/sysctl.h
- driverlib/gpio.h

The “LCD.h” also has the definition of the “LCD” structure, which is used heavily in the implementation of the LCD communication code.

For help with the connections to the LCD we followed the guide found here https://microcontrollerslab.com/16x2-lcd-interfacing-with-tm4c123-tiva-launchpad-keil-uvision/ . The LCD driver we used is based on the open-source driver found at https://github.com/lspaulucio/LCD_Library it was then modified to suit the specific needs of our project. For example, the sendnum function was modified to send 2 digits (which we needed to display the temperatures on the LCD) instead of the default 4.

This is the list of functions that are declared in “LCD.h” and defined in “LCD.c”:

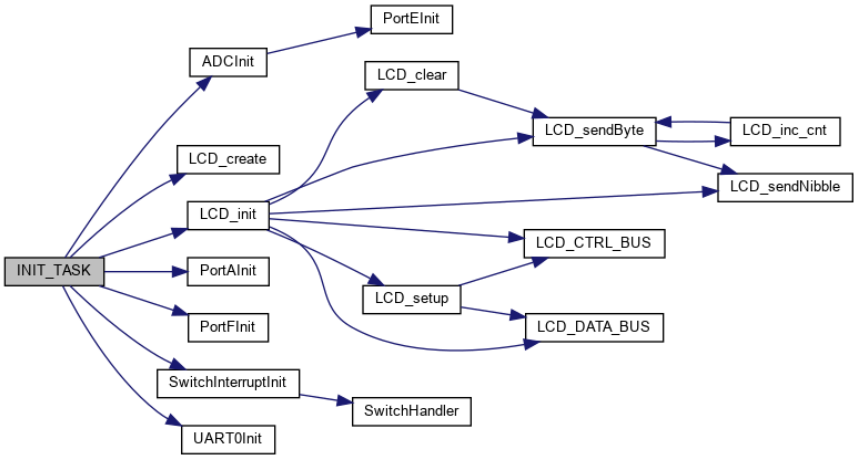
- LCD LCD_create(void): Creates a new LCD structure with the default values.
- uint8_t LCD_CTRL_BUS(LCD *lcd): Returns the value to be written to the control bus using the information from the given LCD structure.
- uint8_t LCD_DATA_BUS(LCD *lcd): Returns the value to be written to the data bus using the information from the given LCD structure.
- void LCD_setup(LCD *lcd): Sets up the LCD of the given LCD structure (enables the peripheral and starts communication).
- void LCD_init(LCD *lcd): Sets up the LCD of the given LCD structure (by calling LCD_setup) and synchronizes/stabilizes the LCD.
- void LCD_clear(LCD *lcd): Clears the LCD screen of the given LCD structure.
- void LCD_sendNibble(LCD *lcd, unsigned char data): Sends the least-significant four bytes of the given character to the LCD screen of the given LCD structure
- void LCD_sendByte(LCD *lcd, bool rs, char byte): Sends the given byte to the LCD screen of the given LCD structure. If rs is set to one, the byte is considered a data byte. Otherwise, it is considered a command byte.
- void LCD_sendNum(LCD *lcd, unsigned int num): Sends the given number (as a string) to the LCD of the given LCD structure.
- void LCD_sendString(LCD *lcd, char* word): Sends the given string to the LCD of the given LCD structure
- void LCD_inc_cnt(LCD *lcd): Moves the cursor of the LCD screen of the given LCD structure to the next character location
- void LCD_setPosition(LCD *lcd, unsigned int line, unsigned int col): Moves the cursor of the LCD screen of the given LCD structure to the given line number and column number.

“main.c”

The “main.c” file has the main code that links together the different libraries to achieve the final product. The “main” function first initializes a semaphore, a mutex, and a queue (with five slots). The semaphore is given when the PortF switches are pressed to change the set point value. The mutex is used for data access safety. The queue is used to place the values from the temperature sensor and holds the values to be sent to be displayed through LCD or console.

The “main” function then creates three tasks (higher priority first):

1. INIT_TASK: Initializes PortA, PortF, UART0, switch interrupts and the LCD. This task runs only once then deletes itself.



2. vChangeThresholdTask: Takes the threshold from the user. This task gets triggered by the switch interrupt triggered by either switch on PortF being pressed through the xBinarySemaphore.
3. vDisplayTask: Used to display the temperature on the LCD. It also converts the temperature from Celsius to Fahrenheit and Kelvin to be displayed over UART. This is a continuous task.
4. vADCTask: Used to get the temperature from the sensor. This is a continuous task.

Project Repository Link:

https://github.com/Mazen-Ghaleb/Real-time-Embedded-Systems

Project Link:

https://drive.google.com/file/d/1hvDvGMmMpE4LZFwXFkgqbeYRAMUERn2x/view?usp=sharing

Video Link

https://drive.google.com/file/d/1bP2qmMBpKowg0wejrJv1Dhp6TNBjxZVH/view?usp=sharing