ECEN 5013: Embedded Software Essentials

Project 2: Microcontroller Timers, UART and Profiling

**Group members:**

Gaurav Gandhi

Sanjana Kalyanappagol

Introduction

The goal of the project is to implement various tasks using Microcontroller timer, UART and profiling. Initially UART logger and controller was implemented in which UART was used with ftoa() function, then UART Logger and LED Controlled UART was designed.

In the next part of the project, Circular buffer was combined with the UART and various unit-test cases were tested.

In the last part of the project, a code profiler was designed to compare to the execution times of various functions with different sizes using Timers and library functions for FRDM board and BBB respectively, also compile time flags were used to customise the implementation.

Sourcecode:

Hardware used:

1x Freedom Freescale Board

1x BeagleBoneBlack Board

A serial terminal program-Putty

**Project Procedure:**

**UART Logger and Controller**

In this a ftoa() function was implemented combined with the configuration of UART interface of the Freedom Freescale board which can print data off on board without the use of the printf and the debug console. To do this, we have used the same USB connector labeled as for the debug interface. This connector has an internal UART to USB converter in the onboard OpenSDA Emulator.

The UART configuration:

•LSB first (on by default)

•1 Start/1 Stop Bits

•8-bit data transfers

•No Parity

•57200 Baud

**Transmisson**: Transmit interrupts was used to send data and offload the work on the main loop.

**Reception**: RX UART interrupts were used.

UART Logger

We have used ftoa/itoa functions to transmit the variable data through the UART. We have written a code inside of some wrapper functions called LOG0() in which we will use to send data to your serial terminal and LOG1() function sends data with parameters. This log function will take a string of ASCII characters to represent on the serial terminal. We have designed an own function to concatenate ASCII strings with the converted numbers.

UART Controlled LED

RGB PWM capable pins on the Freescale board via UART RX were used to change colours and brightness of each colour. Timer routines that can pulse width modulate the RGB LED were used to produce PWM RGB LED functionality in Edge aligned PWM mode. All Colors have an adjustable duty cycle. Meaning, given any runtime input, we are able to adjust the duty cycle outputs i.e the brightness of the LEDs. We have accounted for 8 basic combinations such as Red, Yellow, Green, etc.

In this, certain keyboard characters to “control” the LED functionality were mapped. Such as ‘a’,w’,’s’, & ’d are being used to cycle between colors and brightness.

Input character ‘t’ is being used to be string echo mode where we grab a sequence of characters entered via UART until a newline character is received, once received , it will print out the characters back to the terminal via the Logger.

SCREENSHOTS

DEMO

Circular Buffer and Unit-tests:

Circular Buffer was designed and various functions such as buffer full, buffer empty, add item, read item, initialisation and destroy function was added.

Unit-tests were written to check the circular buffer with various tests such as given below and Unit-test suite was also implemented and tested successfully:

* buffer full returns success/failure for empty/non-empty/full conditions
* buffer empty returns success/failure for empty/non-empty/full conditions
* Add item to the buffer works correctly:
* Should error on overfill
* Should wraparound at the boundary
* Add-item and removing that item at that location should work
* Fill completely–check current items added/size to see this is full
* Remove an item from the buffer (should return error on empty)
* Should error on overempty
* Should wraparound at the boundary
* Add-item and removing that item at that location should work
* Empty completely–check current items added/size to see this is zero.
* Initialize should return a valid pointer of heap when heap is available
* Destroy should free memory back to malloc

S

SCREENSHHOT

**Mallinfo** data structure was used from the malloc library to find information on the current heap allocation.

**Screenshot**

ATTACH CIRCULAR BUFFER TO UART