CE-2812, Lab Week 4, Console Application

Preliminary version. Subject to minor changes / additions / deletions.

1 Purpose

The purpose of this lab is to write a console application tha control. Systems such as this are widely used in embedded systems when in-circuit

2 Prerequisites

- The Nucleo-F446RE board has been mounted onto the Compute Engineeri el d.

3 Activities

3.1 Overview

Create a console application that will present a menu to the user. At a minimum, the following menu options shall be implemented:

Read memory word (RMW) – the user will provide an address and the program will read an unsigned 32-bit word at that address and print the contents of that address to the console in hex and decimal format (optional – also binary). This command should ensure it only reads addresses on a word boundary.

Write memory word (WMW) – the user will provide an address and a value and the program will write the provided value as an unsigned 32-bit word to the provided address. This command should ensure it only writes addresses on a word boundary.

Dump memory (DM) – the user will provide an address and an optional length and the program will dump the contents of that block of memory to the console. If no length is supplied, defaults to 16 bytes. The output should be formatted with 8 or 16 bytes per line, in hex. Each line starts with the address. You may choose to always dump a multiple of 8 or 16 bytes (for example, if user requests 20 bytes, you could display two rows of 16 bytes for a total of 32 bytes displayed. This memory dump should read individual (unsigned) bytes from memory and access each address sequentially.

Help – provides the user with detailed help in using your program.

Be sure to make your program easy to use and intuitive. For example, your will likely want to default to hex notation for addresses, although you can get this behavior automatically by using scanf/strtol properly. You should support either decimal or hex for data values.

Commands issued by the user must be entered and interpreted as a single string. For example, say the user would like to dump 100 bytes of memory starting at address 0x20000000. Your menu system should accept the command:

```
> dm 0x20000000 100
```

as a single input. Your program will then have to parse the command and the arguments from the command and then generate the requested output. **You may not prompt the user for individual inputs.**

You must use provisions within the standard library to parse the input string. One choice is to use [s]scanf(), which we have discussed. Other choices include strtok() and various routines that can search strings, locate sub-strings etc.

You may wish to implement some additional menu items. These are completely optional. Some ideas:

LED Control – write a particular LED or write a particular value to the bank of LEDs.

LCD Control – be able to send messages to LCD and/or control LCD (clear screen, set position, etc.).

Keypad Scan – scan the keypad and return keystrokes to the console.

3.2 Sample Session

The following shows what an interaction with your system may look like. The '>' is a prompt generated by your program. Items in bold are typed by the user.

```
> dm 0x20000000 25
0x20000000: 01 11 1F 0E FF A0 55 80 23 43 E9 CD F0 DE AA 03
0x20000010: 87 14 41 65 21 11 87 DA A2

> rmw 0x20000000
0x20000000: 0x0E1F1101   236916993    0b00001110000111110001000100000001

> wmw 0x20000000 10
0x20000000: 0x0000000A   10   0b00000000000000000000000000001010

> rmw 0x20000000
0x20000000: 0x0000000A   10   0b00000000000000000000000000001010
```

Can print this row complete if you wish as long as it prints at least as many bytes as requested.

Binary is optional.

You may wish to test **rmw** and **wmw** with I/O registers. You should be able to light the LEDs with a correct set of commands.

4 Deliverables

When completed:

1. Submit to Canvas a **single pdf** printout of your completed source code to Canvas. Include in a comment block at the top of your code a summary of your experience with this project.
2. Ask to demo your lab to instructor. You can do this via writing your name on the whiteboard (in-person) in the lab or sending a Teams chat message (remote).
    a. If you demo during lab in Week 4, you will earn a 10% bonus on this lab. ← really not expecting this!!
    b. If you demo during lab in Week 5, you will be eligible for full credit.

- Demos are ONLY accepted during lab periods. If you are unable to demo by the end of lab in Week 5, you lose the 10% of the assignment attributed to the demo (per syllabus).
- Demos must be ready a reasonable amount of time before the end of the lab period. If you write your name on the board at 9:45 and lab ends at 9:50, and there are five names in front of yours, you will be unlikely to complete your demo by the end of lab and hence lose points.