Christopher Andrade, Brayden Lam
May 24, 2024
Lab 5 Report

Procedure

Lab 5 concerned utilizing the CPUlator debugger utilized in Lab 4 to manipulate certain outputs and inputs to/from NIOS peripheral devices. Furthermore the CPUlator utilized both ARM assembly instructions and C code.

Task 1

Part 1

The ”Volatile long*” keyword found at the beginning of the given C code declares a pointer to a “long” 32-bit integer that is volatile, meaning the integer can be changed at any time, without any action being taken by the code.

The NIOS LEDs turn on according to the binary number assigned to it. To turn on the farthest 4 right LEDs you write an F to it. So to turn on all the LEDs it is 0x3FF. The 3 turns on the first 2, the first F turns on the next 4, and the last F turns on the last 4.

Code to illuminate all 10 of the LEDs on the NIOS board below:

```
#define LED ((volatile long *) 0xFF200000)
int main() {
    *LED = 0x3FF;
    return 0;
}
```

Part 2

The PC address of the main branch is 00000230. It is not 0 because the program was loaded into memory starting from a different address from the compiler. The while(1) loop is necessary to keep the program running indefinitely. The program needs to continuously update and without an infinite loop it would check the switches and LEDs once then stop. R2 holds the value of Swval. The values it switches through are FF200000, FF200040 and Swval. FF200000 and FF200040 are the addresses of the LED and Switch register while Swval is the value of the switches.

Task 2

Part 1

The register address for the LEDs on the NIOS processor is 0xFF200000, which can be defined in C as “#define LEDS ((volatile long*) 0xFF200000)”. The register address for the Push Buttons on the NIOS processor is 0xFF200050, which can be defined in C as “#define PUSHBUTTONS ((volatile long *) 0xFF200050)”.

The information corresponding to what push buttons are currently pressed are pointed to by the PUSHBUTTONS pointer and can be accessed by assigning an int to the pointer, which will now hold the value of the push buttons status. The LEDs can be lit by utilizing the LEDS pointer to point to the same int value which the PUSHBUTTONS pointer edits.

The size of the ELF executable generated from the C code implementation is 2656 bytes.

C Code to illuminate LEDs according to push buttons on the NIOS board below:

```
#define PUSHBUTTONS ((volatile long *) 0xFF200050)
#define LEDS ((volatile long*) 0xFF200000)
int main() {
    int butval;
    while (1) {
       butval = *PUSHBUTTONS;
       *LEDS = butval;
    }
    return 0;
}
```

Part 2

The ELF produced by the assembly code was significantly smaller, the ELF executable produced by the C code was 2656 bytes long, while the ELF produced from the given and edited assembly was only 28 bytes, nearly 100 times smaller than the C ELF executable. The ELF executable created by the C code is significantly larger as it contains many other instructions which concern the initialization of the stack pointer. Furthermore, there are multiple branches that are labeled as part of the exiting process that are many instructions long which add to the size of the C ELF executable.

To complete the given Assembly instructions the addresses of the push button and LED values in memory must be provided which are given within the CPUlator debugger.

Assembly Code to illuminate LEDs according to switches on the NIOS board below:

```
.equ PUSHBUTTON, 0xFF200050
.equ LED, 0xFF200000
start:
    movia r2,PUSHBUTTON
    ldwio r3,(r2)     # Read in buttons - active high
    movia r2,LED
    stwio r3,0(r2)     # Write to LEDs
    br start
```

Results

Task 1

Screenshot of the single lit LED on the NIOS board is shown below in figure 1.

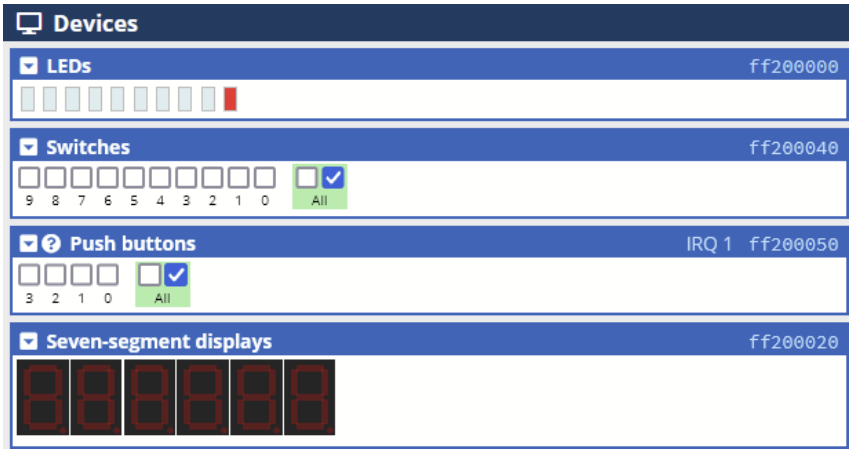


Figure 1: Screenshot of the LEDs on CPUlator, showing that the right-most LED is illuminated.

Task 2

Screenshot of the LEDs on the NIOS board being lit according to the current push button input utilizing C code is shown below in figure 2.

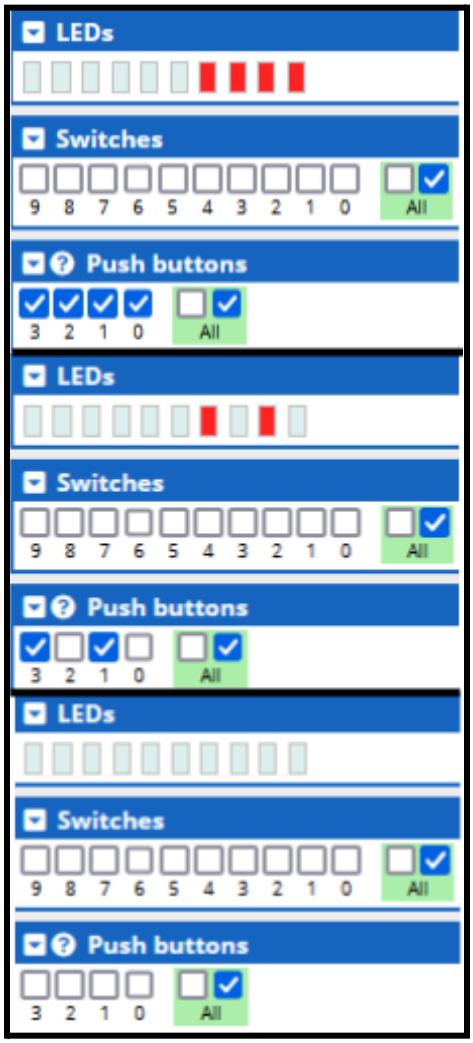


Figure 2: Screenshot of LEDs and different switch combinations on CPUlator, C implementation.

Screenshot of the LEDs on the NIOS board being lit according to the current push button input utilizing ARM instructions is shown below in figure 3.

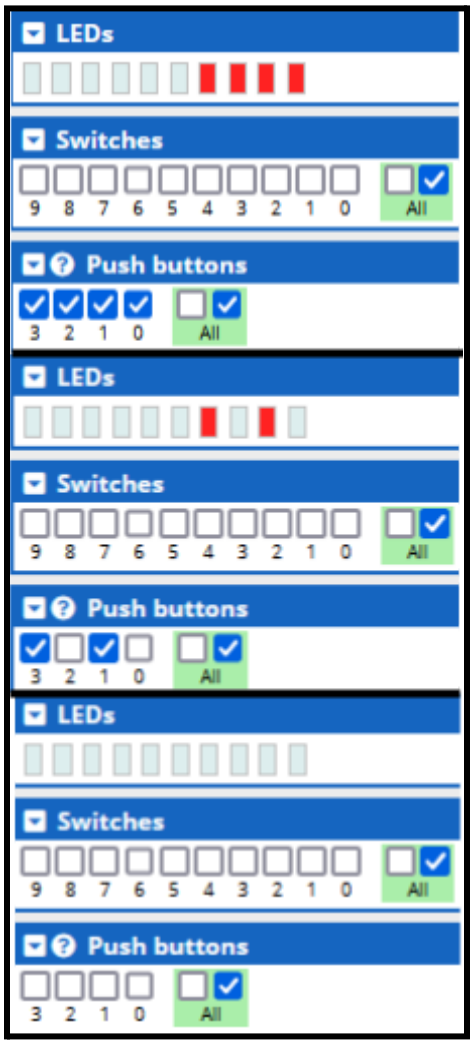
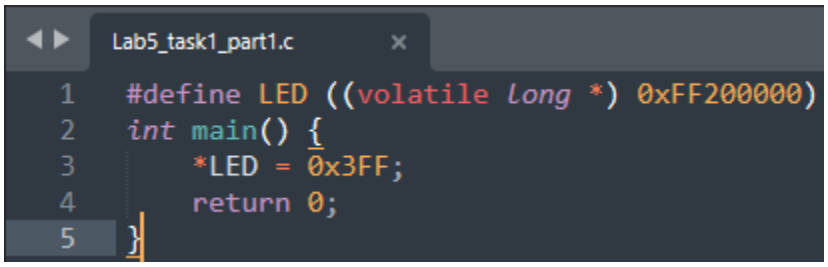


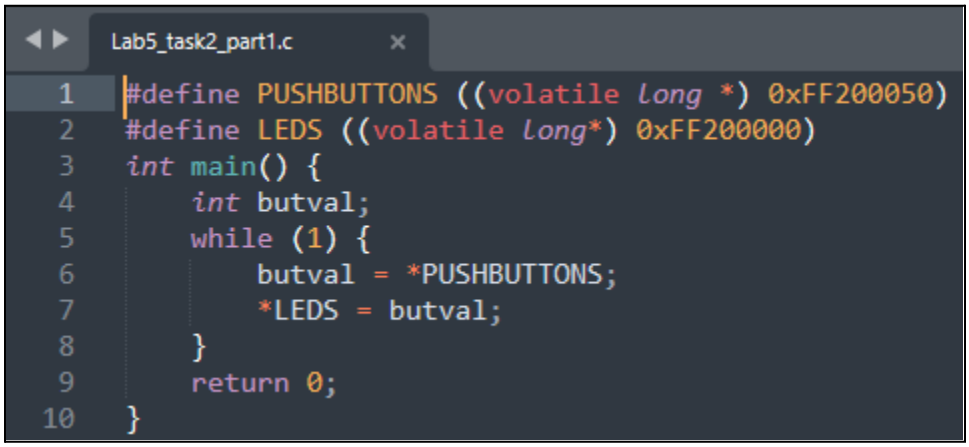
Figure 3: Screenshot of LEDs and different switch combinations on CPUlator, Assembly implementation.

Appendix

Lab5_task1_part1.c



Lab5_task2_part1.c



Lab5_task2_part2.s