Experiment 2 Fall 2012

What's My UIN?

Go over experiment 1 for a complete explanation of delay loops. You will need to implement delays in experiment 2.

Design a system that will output the digits of your UIN in ASCII (i.e. a 7-bit binary number - note that actually only 6 ascii bits are required for decimal numbers since the most significant bit is zero) and in 4-bit binary. Example: for the decimal digit 6 the 10 bit output will be - 0110 110110 (4-bit binary on the left, 6-bit ASCII on the right.)

A START DISPLAY signal is required to begin displaying the digits. The output should be blank until the START DISPLAY signal goes LOW.

When the START DISPLAY signal goes LOW the digits should display one digit at a time (as above.)

After the digits begin being displayed the START DISPLAY signal should be ignored.

Each output digit should display for about 1 second. After the last digit is displayed (for one second) the display should go blank and the system should go into an idle state and do nothing.

The RESET signal of the NANOCORE12 should be implemented so that the program can be re-started.

Design and implement the system with the following specification:

I. Input:

1. One line to RESET the system.

2. One line to implement the START DISPLAY key.

Use the "push button switch" and 10K "Pull-up" resistors that came with your ECE 367 parts kit to create these input lines.
Use PORT M pin PM3 for the START DISPLAY input signal.
Use the NANOCORE12 RESET input to implement the system RESET. There must be a PULL-UP resistor from RESET (pin 30) to Vcc. When pin 30 is connected to ground the NANOCORE12 will re-start running the code from the Start label location.

II. Output:

1. Use the LED BAR GRAPH DISPLAY (with a 470 Ohm current limiting resistor for each LED bar - use the 470 Ohm resistors from your ECE 367 parts kit) to show the BINARY and ASCII output.

   Use PORT T bits , and if necessary, Port M bits PM0-1 for the signals to the LED BAR GRAPH

III. Operation:

1. At power up or RESET pulse (signal that is pulled LOW and then HIGH) the LED's should go blank and wait for the START DISPLAY signal.
2. When the START DISPLAY key goes LOW the UIN digits begin to display in BINARY and ASCII

Use "polling" to check for the START DISPLAY key. Assuming START DISPLAY is held HIGH by the pull-up resistor (switch open) then the following code slice can check for LOW (switch closed)

```
HERE: BRSET PORTM, $08, HERE ; Branch if PM3 is SET (1) to HERE - i.e repeat this line
; i.e check PM3 again. Hence polling! Assumes PORTM = $250 and the DDRM = $252
code to run when PM3 = LOW ; If we get here PM3 is LOW
```

Your UIN number is "HARD CODED." The UIN number is permanently in the code. The only way to change it will be to re-assemble the code after you change those digits.

You MUST use "look-up" tables to store the digits for your UIN. See examples from class.

Experiment 2 is due WEEK 4 (9/18)

Report: Your lab report must include the following:
1. A cover page with: Experiment Name and number, ECE 367 Spring 2012, Your name, Your UIN, the date submitted, and your TA's name.
2. Your assembler code with your name, course, date, experiment number, program explanation and comments for every line of code. The program explanation should include the NanoCore12 pin assignments and an explanation of the organization of the data table.
3. A logic diagram of the circuit
4. An electric circuit diagram of the complete system
5. A user manual to explain how to use the system.
6. Conclusions: Does your circuit meet the specifications and function properly? What problems did you encounter during the coding or construction of the circuit? How would you do this project differently? What extra features or functionality could you have included? What did you learn from working on this project?

Last modified: Mon Sep 10 10:00:44 2012

ECE 367: Experiment 2

Project: What’s My UIN

Semester: Fall 2012

Name: Kai Zhao

UIN: 670720413

Due Date: 2012 Sept 18

Lab Section: T11

TA: Chenjie Tang

User Manual:

To determine what is my UIN, first supply power by USB and press the reset button on the left of the bar graph. Press the start button on the right of the bar graph when ready. To restart, press the reset button and then the start button.

Conclusion:

Yes, my circuit meets the specifications and function properly.

I had trouble figuring out how to poll for the start button. I tried setting the direction of port M to input before polling. However, use the BRSET op-code did not seem to work unless it is the first line of code after the setup code.

I would not do this project differently.

I could have included an optional one button start switch, as opposed to having to use reset and start button. Therefore, the user will be able to start the program with 1 click or 2 clicks, whatever the user prefers.

I learned about program layouts, port memory locations, polling, loop-up tables, push-switches, and 7-segment LED displays.