Lab Assignment 3—*ETHOS*
Designing with I²C-based Input and Output Devices

Overview:

This week you'll create *ETHOS*, the ***E****lectronic* ***T****emperature &* ***H****umidity* ***O****bservation* ***S****ystem.* *ETHOS* will gather temperature and relative humidity information from an Si7021 I²C-based temperature-humidity sensor and continuously display the results on an I²C-based 1602A 16x2 LCD display. Also, *ETHOS* will provide temperature and humidity information to remote users interactively via RS232. Designing and building *ETHOS* will give you an understanding of the I²C protocol and how to interface I²C devices with the NodeMCU microcontroller.

Background:

This lab brings together various communications protocols, sensors, and display devices, so you'll need to become familiar with them before completing this lab. (The links below can be used to understand all these pieces, but they may not be sufficient. You may need to do some of your own hunting to get everything working.)

I²C Protocol
- The I²C bus standard: www.i2c-bus.org
- Learning I²C: www.robot-electronics.co.uk/i2c-tutorial
- Wire Library (w/ I²C) for Arduino: www.arduino.cc/en/Reference/Wire

Adafruit Si7021 Temperature & Humidity Sensor Breakout Board
- Product page: www.adafruit.com/product/3251
- Learning Guide: https://learn.adafruit.com/adafruit-si7021-temperature-plus-humidity-sensor
- Si7021-A20 Datasheet (lab repo)

1602A LCD w/ I²C "Backpack"
- 1602A.pdf (lab repo)
- I²C backpack: www.electroschematics.com/12459/arduino-i2c-lcd-backpack-introductory-tutorial/
- LCD using NodeMCU:
  - https://www.losant.com/blog/how-to-connect-lcd-esp8266-nodemcu
  - Or https://microdigisoft.com/how-to-use-i2c-lcd-with-esp8266-nodemcu-on-arduino-ide/
- More info (but needs some parsing):
  - https://randomnerdtutorials.com/esp32-esp8266-i2c-lcd-arduino-ide/

(NOTE: Using the built-in LiquidCrystal_I2C worked for me, but it does throw a warning that it was only designed for AVR micros…)

Use what you learn from these sources to answer the following questions:

Questions:

1. List the four connections common to any I^2C device.

2. Briefly explain the differences between I^2C "masters" and "slaves."

3. Explain why pull-up resistors are needed on the I^2C data (SDA) and clock (SCL) lines, and why only one pair is needed even though many I^2C devices may be connected to the I^2C bus simultaneously. Also, give typical values for I^2C pull-up resistors.

4. On the NodeMCU's I^2C bus, a logic "1" corresponds to 3.3 V, but on the LCD backpack it corresponds to 5 V. Is this an issue? If not, explain why not. If it is, explain how to resolve it.

5. As you learned from the reading, every I^2C slave device must have a unique address. List the **default slave addresses** of both the **Si7021** and the **LCD backpack**. Also, if the addresses can be changed, list the range of allowable addresses and describe how to change them.

6. List the **range**, **units**, and **accuracies** of the Si7021-A20 temperature and humidity sensor.

7. Given the conversion times of the Si7021, what is the **minimum** amount of time you should allow between readings? Explain you answer.

8. For the 1602A LCD module, list the number of lines that can be displayed, the number of characters per line, and the size (columns x rows in dots) of each character.

9. The 1602A has a built-in character set that includes many ASCII characters, but it contains other symbols as well. Give the **hex character value** for the "degrees" symbol (a small circle that appears in the upper left corner of the character field).

10. List the NodeMCU pins that correspond to the I^2C SCL and SDA signals.

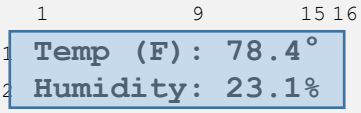
SCL pin: _____

SDA pin: _____

Lab Procedure:

Begin creating *ETHOS* using the following design specifications:

DESIGN SPECIFICATIONS: Once powered up, *ETHOS* must display the current temperature and humidity on the LCD using the following format (small numbers indicate row and column positions):



New temperature and humidity readings must be taken every five seconds and the display updated. There shall be no noticeable LCD screen flicker or "blanking" between readings.

ETHOS must also interact with users via RS232 (9600N81). Use PuTTY with no local echo and no LF added to a CR. If this isn't clear, ***ask your instructor!*** Initially a welcome message must be displayed on the terminal, followed by a command menu. A "Choice? " prompt should appear just below the menu. *ETHOS* must respond to both upper and lower case letters, and *ETHOS* must immediately echo received characters back to PuTTY. When "Enter" is pressed, *ETHOS* must echo back a CR/LF pair and then perform the requested action (e.g. display the temperature). Selecting "Exit" from the terminal menu shall stop all further RS232 communication, but shall not stop the updating of the LCD display.

Figure 2. shows the initial menu along with the interactions involved for each menu choice. **Any input other than what is allowed must result in the menu being displayed again** (i.e. as if the user had selected "?" for Help). *NB:* Your menu, user prompts, and responses must look ***exactly*** like those shown in Figure 2. Also, RS232 interactions with *ETHOS* shall not inhibit or delay the regular reading and displaying of temperature and humidity values on the 1602A LCD. In other words, the system must be highly **responsive**!

Figure 1. *ETHOS* design specifications.

```
Welcome to ETHOS
The Electronic Temperature & Humidity Observation System

Your options:
   T  Request temperature
   H  Request relative humidity
   B  Request both temperature and relative humidity
   ?  Help (shows this menu)
   X  Exit
Choice? t

Temperature: 78 degrees F

Choice? h

Relative humidity: 23%

Choice? B

Temperature: 78 degrees F, Relative humidity: 23%

Choice? ?

Welcome to ETHOS
The Electronic Temperature & Humidity Observation System

Your options:
   T  Request temperature
   H  Request relative humidity
   B  Request both temperature and relative humidity
   ?  Help (shows this menu)
   X  Exit
Choice? x

Goodbye!
```

Figure 2. Terminal screen showing the *ETHOS* menu and remote user interactions.

Turn-In

This is a **two-week** lab. **No formal lab report is required**, but you must turn in:

- **PDF** of this sheet with all questions answered
- a neat and complete schematic of your final hardware design with proper symbols, component values, labels, individual signal lines, power, ground, etc.
- A StateChart (by hand or in any tool) that shows the operational flow of your code
- And, Push your final code back to the GitHub Classroom repo.

In addition, you must demonstrate your working system to your instructor at or before the beginning of lab **two weeks from today.** During the demo, be prepared to answer questions about your design, such as why you chose the pins you did, what challenges you faced, what worked and what didn't, etc. **Have fun!**