# **CS 350 5-1 Milestone Three: Input with Buttons Lab**

# **Austin Donaubauer**

# **Southern New Hampshire University**

# **CS-350 Emerging System Architecture & Technology**

# **Eric Gregori**

# **July 31st, 2025**

# **CS 350 5-1 Milestone Three: Input with Buttons Lab**

Lab Questions

1. Why does the loop that processes the LED blinking need to run in a separate thread?

The loop that processes the LED blinking needs the separate thread to prevent the program’s main thread from being blocked. If the loop was running on the main thread, because it uses the sleep method it would halt all other operations. Using the separate thread allows the main thread to remain available to handle the user input from the button and continue to run the lights morse code message.

1. What is the purpose of returning to the off state after each completed state action?

With how the state machine is defined, where the state can only be transitioned to a new state from the off state, it is necessary to make sure the state is returned to off. The definition like doDot = (off.to(dot) | dot.to(off)) is an example of this where the off state can transition to dot state and then dot state back to off. The definition does not include other transitions to other states. To handle this in the code, it requires calling the transition twice to change the state to dot and then to off.

1. How could you integrate serial communications to facilitate changing the messages available to the program?

Using a serial communication would be useful in getting user input from the console. Adding an additional message option and getting that value from the user in the console, or having a default value available for the new message so that user can still toggle to that option by hitting the button.

1. How could you use the 16x2 display to provide debugging information to the user when they don’t have access to the application console?

Instead of displaying the morse code message on the display, we could output the current state by using the updateScreen method to output. Another option would be to potentially display any error messages, but to do this I would have to utilize error handling like a try-catch block around the operations to catch errors and then use the updateScreen method to display the error. These options could be implemented easily and make it easier to debug this without the console available.