Joseph Chen

Kevin Su

EE 254L

Proposal—

Morse Code Translator

We plan on creating a Morse code translating system that takes a user’s tapping inputs and displays the corresponding letters and symbols on a VGA display. This is done by when the user “taps” or presses a button in either “short” lengths or “long” lengths to act as dots and dashes in Morse code. Because the FPGA board does not have a Morse key and cannot provide a perfect tapping system, we will also utilize a “calibration” module that will measure a user’s tapping lengths, and using the average to determine whether a button press should constitute a dot or dash. Moreover, because Morse code is not a perfect Huffman-coded system, we also need to calibrate for a “wait” time, the period of time between data inputs. For example, using “1” as dot and “0” as dash, the letter A is: 10 and the number 1 is: 10000.

The Morse code translator will utilize a handful of different modules either produced for other EE254 labs, or our own design. First, a module use to decode dot and dashes into respective symbols will be called **letter\_sm**. Another module will be the debouncer, with modifications to allow calibration of dot/dash recognition as well as wait time. Another module will be simple 3-state dot/dash recognition that intakes debouncer outputs and produces either “dot” or “dash” symbols. The final module will be a calibrator that measures user input and produces “waiting” time (how long should the system continue waiting for inputs before resetting and taking in a new symbol?), as well as dot vs. dash recognition that influences the debouncer outputs. Finally, a top module will be used to describe the relation of our modules and produce the desired outputs on the FPGA as well as the VGA displays.

The following is an example of our **letter\_sm** decoding state machine that interprets “dot” and “dash” inputs: