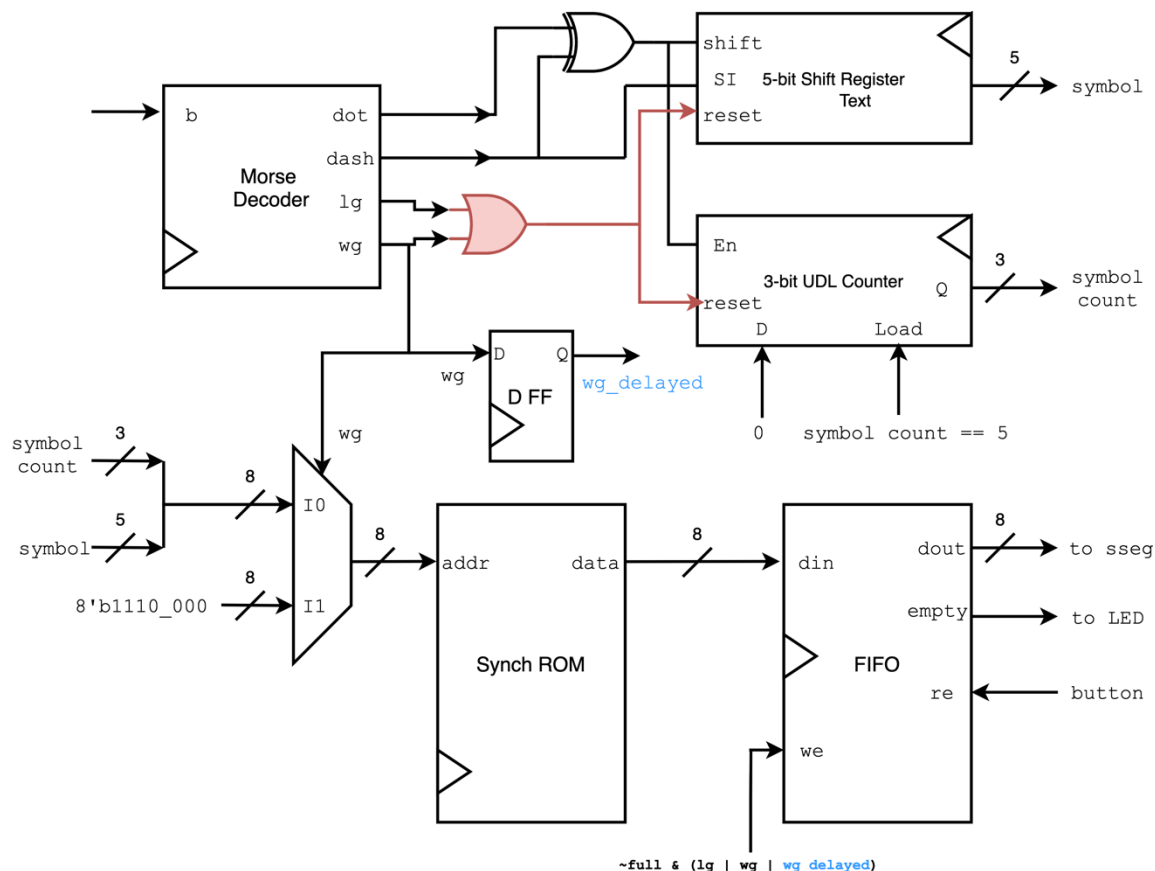# Lab 11
Morse Code to ASCII

In this lab you will use a lookup table (LUT) implemented using a memory elements to convert Morse code patterns of dots and dashes into ASCII code. The outputted ASCII codes will be pushed into a FIFO buffer and displayed on the seven-segment display

- Implement the system shown in the figure on the FPGA board
- The red OR gate resets the 5-bit shift register and the 3-bit UDL counter after each letter (or word)
- `wg_delayed` is necessary to push a space at the end of a word (i.e. after a `wg`)
- Figure out the width and depth of the ROM and FIFO buffer interface
- Use the included `morse2ascii_lut.xlsx` to figure out the initial values of ROM
- You can use your Morse Decoder or the decoder I provided in the starter code. However, if you decided to use mine, please keep in mind:
  - The decoder was designed using a single timer/counter structure
  - The functionality of the decoder depends on knowing when to reset, enable, and disable the timer/counter structure

1

- o You can change the timer value by changing the `TIMER_FINAL_VALUE` parameter
- o Generating a `wg` signal doesn't mean `lg` will also be generated
- o I have tested my decoder; however, I might have missed some corner cases (i.e. use it at your own risk)
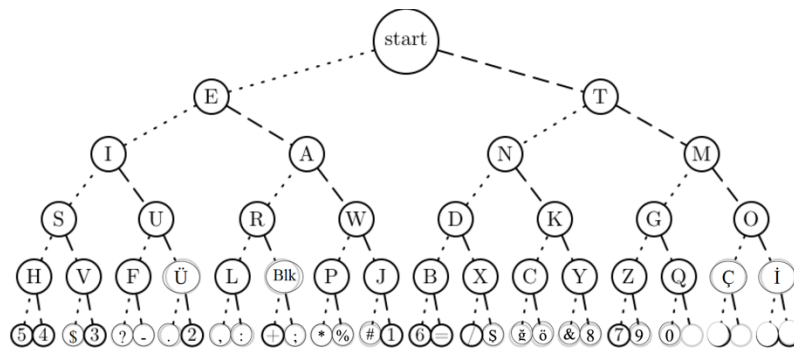
International Morse Code

1. The length of a dot is one unit.
2. A dash is three units.
3. The space between parts of the same letter is one unit.
4. The space between letters is three units.
5. The space between words is seven units.

Note:

I used the binary tree shown above to fill out the lookup table in the excel sheet. To get the letter associated with a certain code, move right for a dash and left for a dot.

Submission check list:

[ ] All Verilog code you generated or modified
[ ] All testbenches written
[ ] Embed all screenshot of your testbench output in your README.md
[ ] Embed all block diagram or state diagrams generated in your README.md
[ ] Short videos demonstrating each of the parts you implemented on the FPGA