Lab3: Morse Code Decoder

National Chiao Tung University
Chun-Jen Tsai
9/27/2016

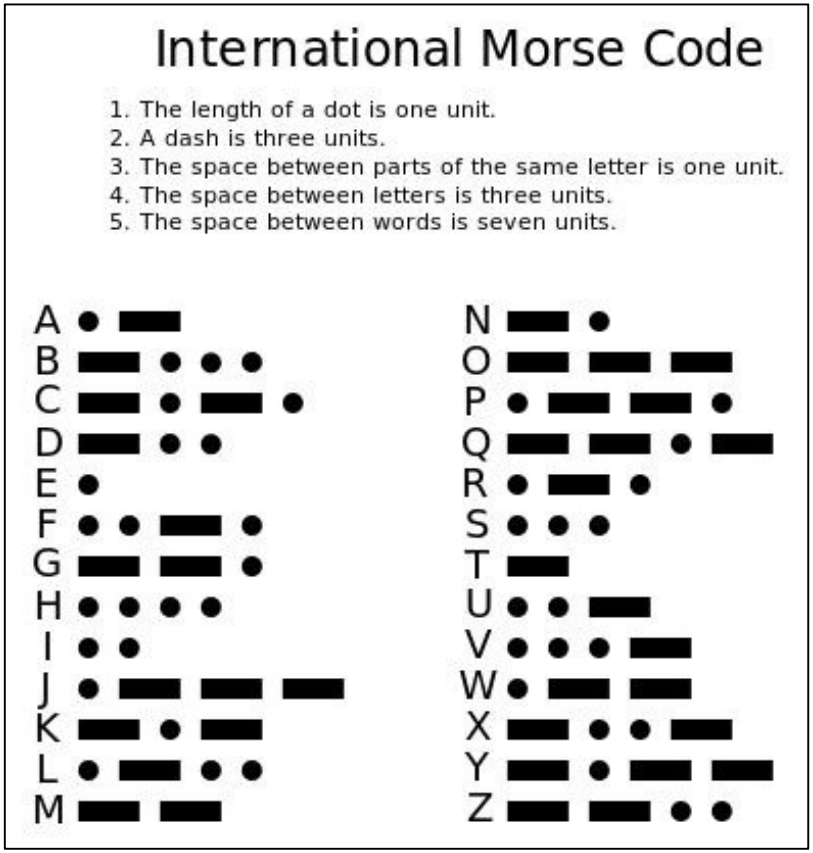
Lab 3: Design a Morse Code Decoder

- In this lab, you will design a Morse code decoder module for a five-letter English word
  - The input is a 80-bit register that contains the Morse code of five English letters
  - The output is a 35-bit register of five 7-bit ASCII codes†
  - The circuit will be tested using a testbench with ISim, you do not have to synthesize the circuit for the Spartan 3 board
- You must demo to your TA during the Lab hours on Oct/11 that your circuit works with ISim

† Note: the ASCII code has only 7 bits, the so called “8-bit ASCII code” shall actually be called “ISO 8859-1 code. ”

Morse Codes for English Alphabet

- In Morse code, there are "on" and "off" signals
  - A dot is one unit of "on"
  - A dash is three units of "on"
  - A space can be one, three, or seven units of "off"
- If we use 0 for "off" and 1 for "on", then:
  - A is 10111,
    B is 111010101,
    C is 11101011101,
    …, etc.
  - The longest letters are composed of 13 bits (J, Q, and Y)



Binary Coding of Morse Code

- The binary Morse codes of A~Z are listed below. Their hexadecimal codes are prefixed with zeros to 16 bits.

A	10111	(0x0017)	
B	111010101	(0x01D5)	
C	11101011101	(0x075D)	
D	1110101	(0x0075)	
E	1	(0x0001)	
F	101011101	(0x015D)	
G	111011101	(0x01DD)	
H	1010101	(0x0055)	
I	101	(0x0005)	
J	1011101110111	(0x1777)	
K	111010111	(0x01D7)	
L	101110101	(0x0175)	
M	1110111	(0x0077)	
N	11101	(0x001D)	
O	11101110111	(0x0777)	
P	10111011101	(0x05DD)	
Q	1110111010111	(0x1DD7)	
R	1011101	(0x005D)	
S	10101	(0x0015)	
T	111	(0x0007)	
U	1010111	(0x0057)	
V	101010111	(0x0157)	
W	101110111	(0x0177)	
X	11101010111	(0x0757)	
Y	1110101110111	(0x1D77)	
Z	11101110101	(0x0775)	

- The ASCII codes of A~Z are `0x41~0x5A`, respectively.

Decoder Module Specification

- The input/output ports of the multiplier is as follows:

```
module DecodeMorse(
    input clk,
    input enable,
    input reg [79:0] in_bits,
    output reg [34:0] out_text,
    output valid);
```

‘clk’ is the system clock,
‘enable’ raise it to ‘1’ activates the decoding operation,
‘in_bits’ is the input Morse code bitstream,
‘out_text’ is the 7-bit ASCII codes of the output word,
‘valid’ will be raised to ‘1’ when the output is valid

Decoding Behavior

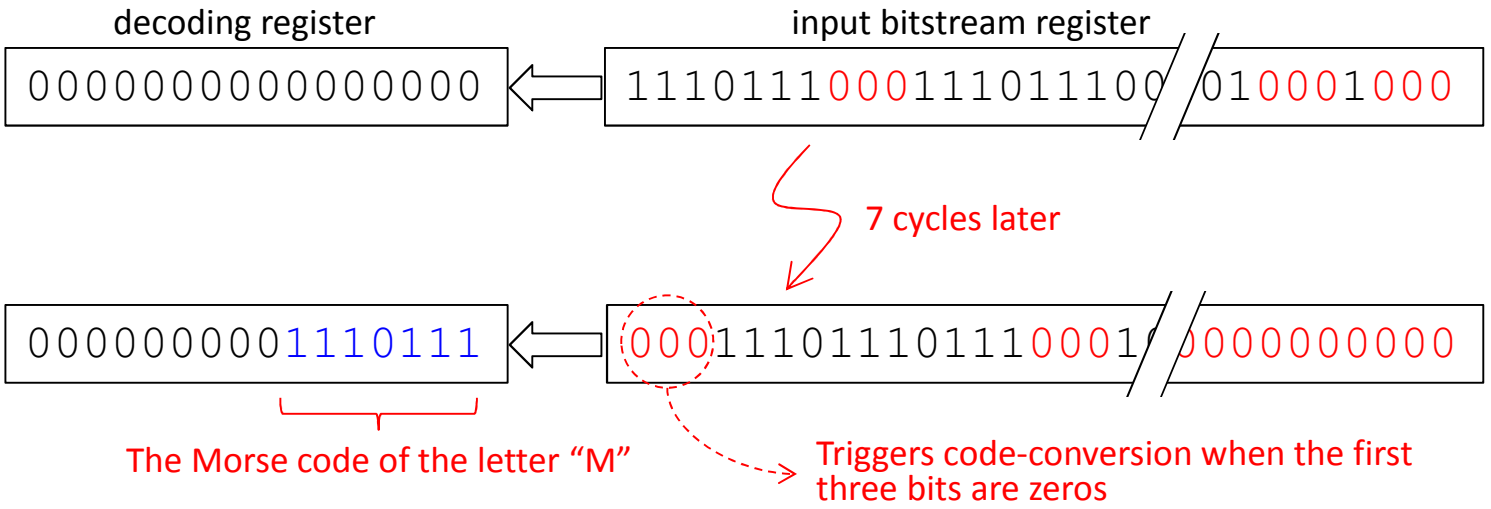
- The input register, `in_bits`, has 80 bits because a five-letter Morse can have up to 80 bits: (13 bits of code + 3 zero bits) × 5 = 80
- You can shift `in_bits` into a 16-bit register, say `dec_reg`, one bit per clock cycle until you see three zero bits at the most significant bits of `in_bits`, in which case `dec_reg` contains the binary Morse code of a letter
- The most challenging task in this lab is to design an efficient code conversion circuit to convert the binary Morse code to a 7-bit ASCII code

Example of Decoding “MORSE”

- The five-letter word “MORSE” in binary Morse code has 46 bits:

  1110111000111011101110001011101000101010001000

- The decoding process involves shifting the input bitstream into a 16-bit decoding register:



Comments on Lab3

- The simplest way to do code conversion is to use a long list of if-then-else statements in Verilog
  → it should work, but not a good coding style

- Please try to use the knowledge you have learned from Mano's book† to design a better way to do code conversion (*Hint: you may want to review Chapter 3*)

† M. Morris Mano, Michael D. Ciletti, Digital Design with an Introduction to the Verilog HDL, 5th Edition, Pearson, 2013.