CS 250 Spring 2017 Homework 03 SOLUTION
Due 11:58pm Wednesday, February 01, 2017
Submit your typewritten file in PDF format to Blackboard.

1. If propagation delay in a combinatorial circuit is measured in gate delays, how long before all outputs are valid for a 16-bit ripple carry adder circuit?
   Answer: 33 gate delays. The general formula is 1 gate delay plus 2 gate delays per bit to allow for the carry ripple to the next more significant bit position. You can check this against the presentation for the 4-bit ripple carry adder in the lecture slides.

2. The 74163 rising-edge-triggered counter chip is being clocked with an SR latch, the same as in Lab 02.
   a. What is the shortest sequence of Set and Reset operations that will advance the count from 14 to 3? Use S to mean set and R to mean reset and write your answer in the form of an ASCII character string.

      Answer: To increment from 14 to 3, modulo 16, requires X = 5 (mod 16) rising edges to be detected by the 74163. The least value for X satisfying the equation is X=5. The SR latch produces a rising edge when it transitions from Q(t) = 0 to Q(t+1) = 1, which happens when the latch transitions from the reset state to the set state.

      What this question is really asking for is ***a program to make the SR latch compute the sequence needed to drive the 74163 count from 14 to 3***. A good first step in any program is to be sure that variables have been initialized if necessary. We are trying to use the latch to compute (to output) rising edges. If the latch is initially in the reset state (Q(t) = 0) we could immediately command Set to compute the first rising edge of the X=5 rising edges needed. However, the latch may initially be in the set state (Q(t)=1) and a Set command would not generate an edge but, rather, sustain Q(t)=1.

      To remedy this inability to use Set to compute a rising edge, we can initialize the latch to the reset state by making the first line of code in our program be a reset command. This line of code with either compute Q(t)=0=Q(t+1), which means sustain computing 0 (no edge in the output) or compute Q(t)=1 → Q(t+1)=0 which means produce a falling edge in the output. Fortunately, a falling edge does not affect the count displayed by the 74163. With the SR latch safely reset, we can compute the 5 rising edges needed by a sequence of lines of code commanding set, reset repeated a total of five times.

      The question asks that our program be written as a sequence of the two ASCII characters R, to mean reset the latch, and S, to mean set the latch. Therefore, the source code is: RSRSRSRSRS.

      If we had coding style standards for readability, we might write

      ```
      R    // initialize Q(t) to 0 in case Q(t-1) = 1
      S    // compute a rising edge by setting Q(t+1) = 1
      R    // make Q(t+2) = 0 in preparation to compute the second rising edge
      S    // compute second rising edge by setting Q(t+3) = 1
      R    // prepare for third rising edge
      S    // third rising edge computed, Q(t+5) = 1
      ```

R // get ready for fourth rising edge
S // fourth rising edge, Q(t+7) = 1
R // final time we need to reset
S // 5th and final rising edge computed

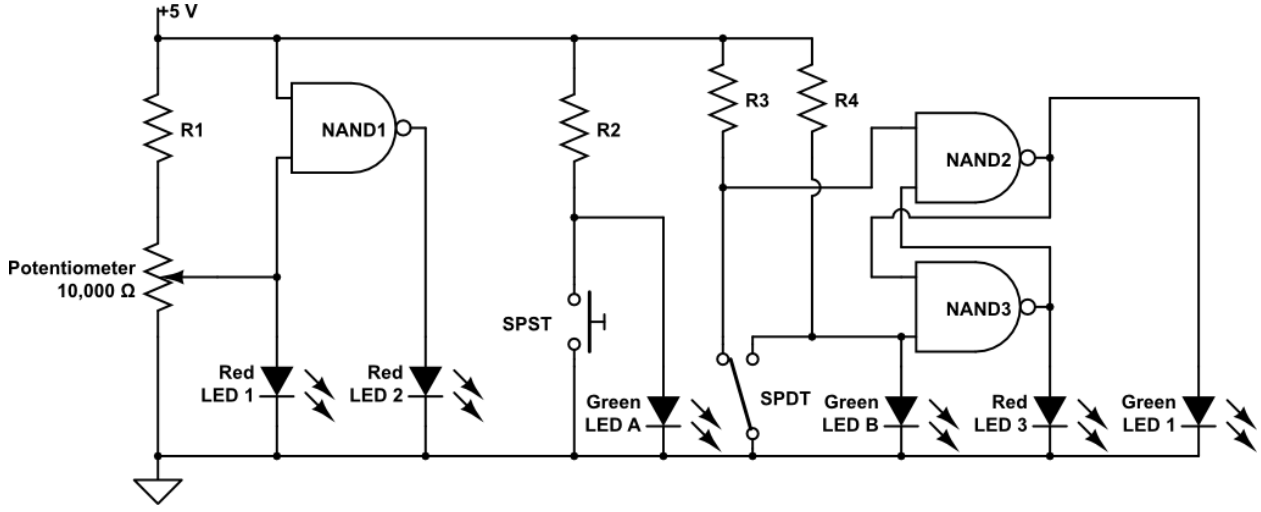
A terminating NULL character is not germain to this ASCII string , unlike in C programming, because the SR latch cannot understand the meaning of NULL. Rather we simply stop providing commands to the SR latch and it will maintain its last Q output for as long as the latch continues to receive electrical power. Maintaining Q means that the count will stop advancing, so our program is a success.

RSRSRSRSRS is your first *assembly language* program of CS250!

b. Extra thought: Using a regular expression, describe all sequences of Set and Reset that will drive the count from 14 to 3.
Answer: Longer sequences that increment the counter by additional multiples of 16 will also drive the count from 14 to 3. One way to write such sequences is RSRSRSRSRS(RSRSRSRSRSRSRSRSRSRSRSRSRSRS)* where the * means replicate the string in parenthesis zero or more times.

Mathematically, the set of all strings that drive the count from 14 to 3 form an *equivalence class* modulo 16 and these strings are all, pairwise, *congruent* modulo 16 with each other. Modular arithmetic is often used to compute checksums that are embedded within information, such as bank account numbers, to detect errors such as transposition of digits when typing the account number into a web form.

3. Consider the schematic below for the following questions. The difference between this circuit and the circuit of Lab 02 is that the 74163 counter has been removed.



a. Which LED shows that it is possible to operate LEDs at half brightness rather than just fully off or fully on. Be sure to use the exact name shown in the schematic so that there is no ambiguity in your answer.

Answer: Red LED1

b. Name all LEDs in the schematic above that are connected to de-bounced clock signals. If no such LED exists, write "None." Be sure to use the exact name shown in the schematic so that there is no ambiguity in your answer.

Answer: Red LED3 and Green LED1

c. Let Red LED3 be the Q' output of the S'R' latch formed by NAND2 and NAND3. The moving pole of the SPDT switch has three positions: connected to R3 (R3, for a short name), in between the R3 and R4 contacts (B, for a short name) where the pole is connected to nothing, and connected to R4 (just called R4). Fill in the five missing entries in the following table for each time step from 0 to 5.

Time	SPDT position	Red LED 3 state
0	R3	OFF
1	R3	**OFF**
2	B	**OFF**
3	R4	**ON**
4	B	**ON**
5	**R3**	OFF

4. Complete the table to show how the given binary strings are writing in each representational form. If a binary string is not valid for a given representation, write "error" in the table.

Given binary string	**Written as octal**	**Written as hexadecimal (0x)**
110011011111	6337	CDF
010111110001	2761	5F1

5. Complete the table to show how the given binary strings are interpreted in each data representation. For numerical representations write your answer in the form of a decimal number. Use care when writing a decimal number equivalent to show a sign when there is ambiguity if a sign is not shown. The table headings "1's" and "2's" are short for one's complement and two's complement, respectively. If a binary string is not valid for a given representation, write "error" in the table.

Given binary string	**Unsigned integer**	**Sign magnitude**	**1's**	**2's**	**ASCII character**
10110100	180	-52	-75	-76	error
00000101	5	5	5	5	enq

00000000	0	+0	+0	0	nul
11111111	255	-127	-0	-1	error

6. What is the 16-bit representation of the 2's complement number 11101010?
   Answer: 1111111111101010. To expand a 2's complement number into a representation having a greater number of bits, simply copy the sign bit into all the additional bit positions.