1. We have several addressing modes to access memory (immediate not listed):
   - Base displacement addressing: Adds an immediate to a register value to create a memory address (used for lw, lb, sw, sb)
   - PC-relative addressing: Uses the PC and adds the immediate value of the instruction (multiplied by 2) to create an address (used by branch and jump instructions)
   - Register Addressing: Uses the value in a register as a memory address (jr)

   a. What is range of 32-bit instructions that can be reached from the current PC using a branch instruction?

   The immediate field of the branch instruction is 12 bits. This field only references addresses that are divisible by 2, so the immediate is multiplied by 2 before being added to the PC. Thus, the branch immediate can move the reference 2-byte instructions that are within $[-2^{11},2^{11}-1]$ instructions of the current PC. The instructions we use, however, are 4 bytes so they reside at addresses that are divisible by 4 not 2. Therefore, we can only reference half as many 4-byte instructions as before, and the range of 4-byte instructions is $[-2^{10},2^{10}-1]$

   b. What is the range of 32-bit instructions that can be reached from the current PC using a jump instruction?

   The immediate field of the jump instruction is 20 bits. Similar to above, this immediate is multiplied by 2 before added to the PC to get the final address. Since the immediate is signed, the range of 2byte instructions that can be referenced is $[-2^{19},2^{19}-1]$. As we actually want the number of 4-byte instructions, we actually can reference those within $[-2^{18},2^{18}-1]$ instructions of the current PC.

2. Given the following RISC-V code (and instruction addresses), fill in the blank fields for the following instructions (you'll need your RISC-V green card!).

0x002cff00: loop: sub t1, t2, t0 |__0x20__|__5_|__7__|__0__|__6__|__0x33__|
0x002cff04: jal ra, foo |__0__|__0x14__|__0__|__0_|__1__|__0x6F__|
0x002cff08: bge t1, zero, loop |__1_|_0x3F_|__0_|__6__|__0xC |_1_|__0x63__|
...
0x002cff2c: foo: jr ra ra=____0x002cff08_____

3. Which step in CALL resolves relative addressing? Absolute addressing?

Assembler, Linker.

4. What does RISC stand for? How is this related to pseudo instructions?

Reduced Instruction Set Computing. Minimal set of instructions leads to many lines of code. Pseudo instructions are more complex instructions intended to make assembly programming easier for the coder. These are converted to TAL by the assembler.

5. Create an XOR gate using only NAND gates.

F=A XOR B= AB'+A'B=((AB')'(A'B)')'

Therefore, five NAND gates are required to create an XOR.

B'=NAND(B,1)
A'=NAND(A,1)
AB'=NAND(A,NAND(B,1))
A'B=NAND(B,NAND(A,1))
F=NAND(NAND(A,NAND(B,1), NAND(B,NAND(A,1)))

6. How many different two-input logic gates can there be? How many n-input logic gates?

A truth table with n inputs has 2^n rows. Each logic gate has a 0 or a 1 at each of these rows. Imagining a function as a 2^n-bit number, we count 2^{2^n} total functions, or 16 in the case of n = 2.

7. Boolean Algebra

$1 + A = 1$ $A + \bar{A} = 1$ $A + AB = A$ $(A + B)(A + C) = A + BC$

$0B = 0$ $B\bar{B} = 0$ $A + \bar{A}B = A + B$

DeMorgan's Law: $\overline{AB} = \bar{A} + \bar{B}$ $\overline{A+B} = \bar{A}\bar{B}$

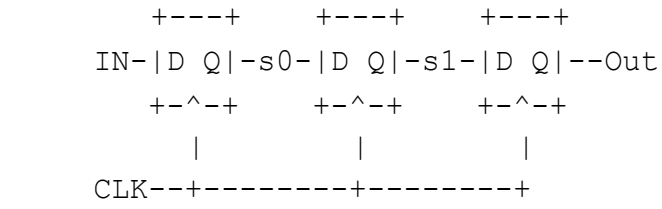
Minimize the following boolean expressions:

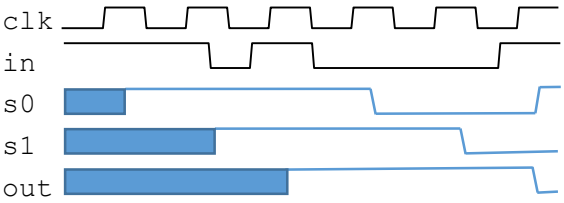
a. Standard: $(A + B)(A + \bar{B})(\bar{B} + C) = A\bar{B} + AC$

b. Grouping & Extra Terms: $\bar{A}\bar{B}\bar{C} + \bar{A}BC + AB\bar{C} + A\bar{B}\bar{C} + ABC + A\bar{B}C = A + \bar{B}\bar{C} + BC$

c. DeMorgan's: $\overline{A(\bar{B}\bar{C} + BC)} = A + B\bar{C} + \bar{B}C$

8. Fill out the timing diagram for the circuit below:

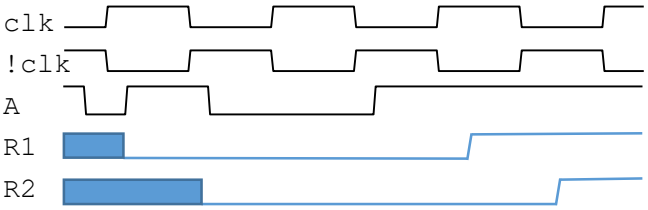




(Delay one cycle)

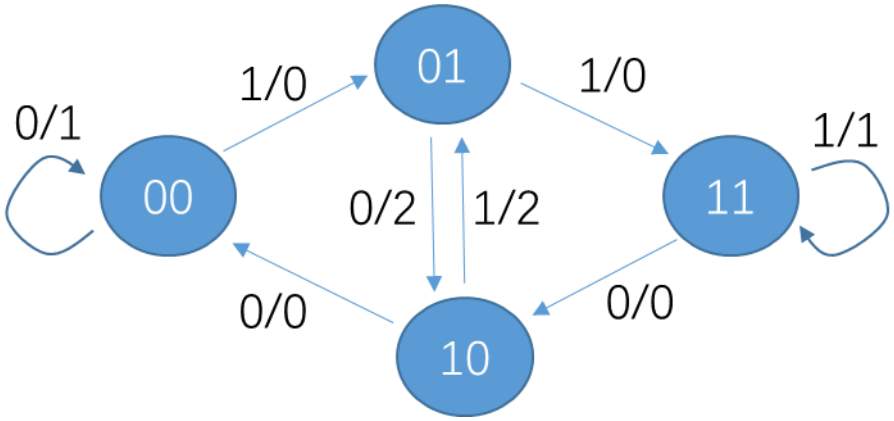
9. Fill out the timing diagram for the circuit below:

```
   +---+     +---+
A--|D Q|-R1-|D Q|-R2--
   +-^-+     +-^-+
     |         |
CLK--+---|>o--+
```



(Delay half cycle)

10. Draw an FSM for outputting a 1 whenever we have three repeating bits as the most recent bits, a 2 whenever we have 010 or 101, and a 0 otherwise. You may not need all states.



00: the most recent two bits are 00
01: the most recent two bits are 01
10: the most recent two bits are 10
11: the most recent two bits are 11