Let L be an arbitrary regular language over the alphabet $\Sigma = \{0, 1\}$. Prove that the following languages are also regular. (You probably won't get to all of these.)

1. $\textsc{FlipOdds}(L) := \{\mathit{flipOdds}(w) \mid w \in L\}$, where the function *flipOdds* inverts every odd-indexed bit in w. For example:

$$\mathit{flipOdds}(\texttt{0000111101010101}) = \texttt{1010010111111111}$$

Solution: Let $M = (Q, s, A, \delta)$ be a DFA that accepts L. We construct a new DFA $M' = (Q', s', A', \delta')$ that accepts $\textsc{FlipOdds}(L)$ as follows.

Intuitively, M' receives some string $\mathit{flipOdds}(w)$ as input, restores every other bit to obtain w, and simulates M on the restored string w.

Each state (q, flip) of M' indicates that M is in state q, and we need to flip the next input bit if $\mathit{flip} = \textsc{True}$

$$Q' = Q \times \{\textsc{True}, \textsc{False}\}$$
$$s' = (s, \textsc{True})$$
$$A' =$$
$$\delta'((q, \mathit{flip}), a) =$$

■

2. $\textsc{UnflipOdd1s}(L) := \{w \in \Sigma^* \mid \mathit{flipOdd1s}(w) \in L\}$, where the function *flipOdd1* inverts every other 1 bit of its input string, starting with the first 1. For example:

$$\mathit{flipOdd1s}(\texttt{0000}\underline{\texttt{1}}\texttt{1}\underline{\texttt{1}}\texttt{10}\underline{\texttt{1}}\texttt{010}\underline{\texttt{1}}\texttt{01}) = \texttt{0000}\underline{\texttt{0}}\texttt{1}\underline{\texttt{0}}\texttt{10}\underline{\texttt{0}}\texttt{010}\underline{\texttt{0}}\texttt{01}$$

Solution: Let $M = (Q, s, A, \delta)$ be a DFA that accepts L. We construct a new DFA $M' = (Q', s', A', \delta')$ that accepts $\textsc{UnflipOdd1s}(L)$ as follows.

Intuitively, M' receives some string w as input, flips every other 1 bit, and simulates M on the transformed string.

Each state (q, flip) of M' indicates that M is in state q, and we need to flip the next 1 bit of and only if $\mathit{flip} = \textsc{True}$.

$$Q' = Q \times \{\textsc{True}, \textsc{False}\}$$
$$s' = (s, \textsc{True})$$
$$A' =$$
$$\delta'((q, \mathit{flip}), a) =$$

■

3. $\textsc{FlipOdd1s}(L) := \{\mathit{flipOdd1s}(w) \mid w \in L\}$, where the function *flipOdd1* is defined as in the previous problem.

Solution: Let $M = (Q, s, A, \delta)$ be a DFA that accepts L. We construct a new **NFA** $M' = (Q', s', A', \delta')$ that accepts $\textsc{FlipOdd1s}(L)$ as follows.

Intuitively, M' receives some string $\mathit{flipOdd1s}(w)$ as input, ***guesses*** which 0 bits to restore to 1s, and simulates M on the restored string w. No string in $\textsc{FlipOdd1s}(L)$ has two 1s in a row, so if M' ever sees 11, it rejects.

Each state (q, flip) of M' indicates that M is in state q, and we need to flip a 0 bit before the next 1 if $\mathit{flip} = \textsc{True}$.

$$Q' = Q \times \{\textsc{True}, \textsc{False}\}$$
$$s' = (s, \textsc{True})$$
$$A' =$$
$$\delta'((q, \mathit{flip}), a) =$$

■

4. $\textsc{Faro}(L) := \left\{ \mathit{faro}(w, x) \;\middle|\; w, x \in L \text{ and } |w| = |x| \right\}$, where the function *faro* is defined recursively as follows:

$$\mathit{faro}(w, x) := \begin{cases} x & \text{if } w = \varepsilon \\ a \cdot \mathit{faro}(x, y) & \text{if } w = ay \text{ for some } a \in \Sigma \text{ and some } y \in \Sigma^* \end{cases}$$

For example, $\mathit{faro}(\texttt{0001101}, \texttt{1111001}) = \texttt{01010111100011}$. (A "faro shuffle" splits a deck of cards into two equal piles and then perfectly interleaves them.)

Solution: Let $M = (Q, s, A, \delta)$ be a DFA that accepts L. We construct a DFA $M' = (Q', s', A', \delta')$ that accepts $\textsc{Faro}(L)$ as follows.

Intuitively, M' reads the string $\mathit{faro}(w, x)$ as input, splits the string into the subsequences w and x, and passes each of those strings to an independent copy of M.

Each state $(q_1, q_2, \mathit{next})$ indicates that the copy of M that gets w is in state q_1, the copy of M that gets x is in state q_2, and *next* indicates which copy gets the next input bit.

$$Q' = Q \times Q \times \{1, 2\}$$
$$s' = (s, s, 1)$$
$$A' =$$
$$\delta'((q_1, q_2, \mathit{next}), a) =$$

■