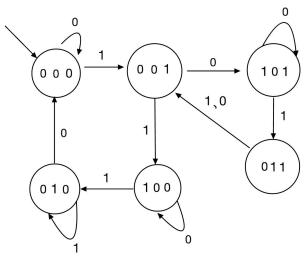


Challenge 2: Truth Tables

Convert the following Finite State Machine into a truth table. This FSM transitions between states based on input *i*, which can be 0 or 1. When constructing the truth table, encode the states in alphabetical order. (Version 1)

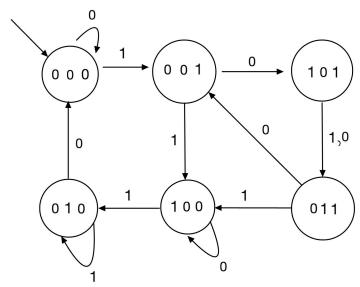


| Inputs | | | | Outputs | | | |
|--------|----|----|---|---------|-----|-----|--|
| s0 | s1 | s2 | i | s*0 | s*1 | s*2 | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |



Challenge 2: Truth Tables

Convert the following Finite State Machine into a truth table. This FSM transitions between states based on input *i*, which can be 0 or 1. When constructing the truth table, encode the states in alphabetical order. (Version 2)

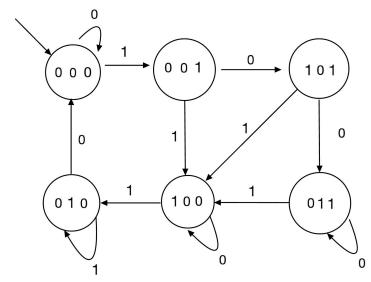


| Inputs | | | | Outputs | | | |
|--------|----|----|---|---------|-----|-----|--|
| s0 | s1 | s2 | i | s*0 | s*1 | s*2 | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |



Challenge 2: Truth Tables

Convert the following Finite State Machine into a truth table. This FSM transitions between states based on input *i*, which can be 0 or 1. When constructing the truth table, encode the states in alphabetical order. Version 3



| Inputs | | | | Outputs | | | |
|--------|----|----|---|---------|-----|-----|--|
| s0 | s1 | s2 | i | s*0 | s*1 | s*2 | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |



Challenge 2 (V1):

*For the actual final, organize your inputs so that i is on the very left. Although, the main idea is that you organize your inputs like you're counting.

| | Inp | uts | Outputs | | | |
|----|-----|-----|---------|-----|-----|-----|
| s0 | s1 | s2 | i | s*0 | s*1 | s*2 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 | 1 | 1 |



Challenge 2 (V2):

*For the actual final, organize your inputs so that *i* is on the very left. Although, the main idea is that you organize your inputs like you're counting.

| Inputs | | | | Outputs | | | |
|--------|----|----|---|---------|-----|-----|--|
| s0 | s1 | s2 | i | s*0 | s*1 | s*2 | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 | |
| 0 | 0 | 1 | 0 | 1 | 0 | 1 | |
| 0 | 0 | 1 | 1 | 1 | 0 | 0 | |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | |
| 0 | 1 | 0 | 1 | 0 | 1 | 0 | |
| 0 | 1 | 1 | 0 | 0 | 0 | 1 | |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 | |
| 1 | 0 | 0 | 1 | 0 | 1 | 0 | |
| 1 | 0 | 1 | 0 | 0 | 1 | 1 | |
| 1 | 0 | 1 | 1 | 0 | 1 | 1 | |



Challenge 2 (V3):

*For the actual final, organize your inputs so that *i* is on the very left. Although, the main idea is that you organize your inputs like you're counting.

| Inputs | | | | Outputs | | | |
|--------|----|----|---|---------|-----|-----|--|
| s0 | s1 | s2 | i | s*0 | s*1 | s*2 | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 | |
| 0 | 0 | 1 | 0 | 1 | 0 | 1 | |
| 0 | 0 | 1 | 1 | 1 | 0 | 0 | |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | |
| 0 | 1 | 0 | 1 | 0 | 1 | 0 | |
| 0 | 1 | 1 | 0 | 0 | 1 | 1 | |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 | |
| 1 | 0 | 0 | 1 | 0 | 1 | 0 | |
| 1 | 0 | 1 | 0 | 0 | 1 | 1 | |
| 1 | 0 | 1 | 1 | 1 | 0 | 0 | |