

# One Hot Encoding

---

# Sequential Design So Far

- \* Designing circuits from state diagrams has been achieved with binary encoding so far
  - Each state is given a binary number
  - Use state table and k-maps to get simplified logic circuits for next states and outputs
- \* Limited by the number of total states and inputs
  - K-map becomes too large to handle by hand

|  | $\overline{a}\overline{b}\overline{c}\overline{d}$ $\overline{a}\overline{b}c\overline{d}$ $\overline{a}b\overline{c}\overline{d}$ $\overline{a}bc\overline{d}$ $a\overline{b}\overline{c}\overline{d}$ $a\overline{b}c\overline{d}$ $ab\overline{c}\overline{d}$ $abc\overline{d}$ $\overline{a}\overline{b}\overline{c}d$ $\overline{a}\overline{b}cd$ $a\overline{b}\overline{c}d$ $a\overline{b}cd$ $\overline{a}b\overline{c}d$ $\overline{a}bcd$ $ab\overline{c}d$ $abcd$ |     |     |     |     |     |     |     |     |     |     |     |     |     |     |   |
|--|---|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|---|
| $\overline{e}\overline{f}\overline{g}\overline{h}$ | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0 |
| 0  | 1   | 5   | 4   | 20  | 21  | 17  | 16  | 80  | 81  | 85  | 84  | 68  | 69  | 65  | 64  |   |
| $\overline{e}\overline{f}\overline{g}h$            | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0 |
| 2  | 3   | 7   | 6   | 22  | 23  | 19  | 18  | 82  | 83  | 87  | 86  | 70  | 71  | 67  | 66  |   |
| $\overline{e}\overline{f}g\overline{h}$            | 0   | 0   | 1   | 1   | 1   | 0   | 0   | 1   | 0   | 0   | 1   | 1   | 1   | 1   | 0   | 0 |
| 10   | 11  | 15  | 14  | 30  | 31  | 27  | 26  | 90  | 91  | 95  | 94  | 78  | 79  | 75  | 74  |   |
| $\overline{e}\overline{f}gh$                       | 0   | 0   | 1   | 1   | 1   | 0   | 0   | 1   | 0   | 0   | 1   | 1   | 1   | 1   | 0   | 0 |
| 8  | 9   | 13  | 12  | 28  | 29  | 25  | 24  | 88  | 89  | 93  | 92  | 76  | 77  | 73  | 72  |   |
| $\overline{e}fg\overline{h}$                       | 0   | 0   | 1   | 1   | 1   | 0   | 0   | 1   | 0   | 0   | 1   | 1   | 1   | 1   | 0   | 0 |
| 40   | 41  | 45  | 44  | 60  | 61  | 57  | 56  | 120 | 121 | 125 | 124 | 108 | 109 | 105 | 104 |   |
| $\overline{e}fgh$                                  | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0 |
| 42   | 43  | 47  | 46  | 62  | 63  | 59  | 58  | 122 | 123 | 127 | 126 | 110 | 111 | 107 | 106 |   |
| $e\overline{f}\overline{g}\overline{h}$            | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0 |
| 34   | 35  | 39  | 38  | 54  | 55  | 51  | 50  | 114 | 115 | 119 | 118 | 102 | 103 | 99  | 98  |   |
| $e\overline{f}g\overline{h}$                       | 0   | 0   | 1   | 1   | 1   | 0   | 0   | 1   | 0   | 0   | 1   | 1   | 1   | 1   | 0   | 0 |
| 32   | 33  | 37  | 36  | 52  | 53  | 49  | 48  | 112 | 113 | 117 | 116 | 100 | 101 | 97  | 96  |   |
| $e\overline{f}gh$                                  | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0 |
| 160  | 161   | 165 | 164 | 180 | 181 | 177 | 176 | 240 | 241 | 245 | 244 | 228 | 229 | 225 | 224 |   |
| $\overline{e}fgh$                                  | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0 |
| 162  | 163   | 167 | 166 | 182 | 183 | 179 | 178 | 242 | 243 | 247 | 246 | 230 | 231 | 227 | 226 |   |
| $e\overline{f}g\overline{h}$                       | 0   | 0   | 1   | 1   | 1   | 0   | 0   | 1   | 0   | 0   | 1   | 1   | 1   | 1   | 0   | 0 |
| 170  | 171   | 175 | 174 | 190 | 191 | 187 | 186 | 250 | 251 | 255 | 254 | 238 | 239 | 235 | 234 |   |
| $\overline{e}fgh$                                  | 0   | 0   | 1   | 1   | 1   | 0   | 0   | 1   | 0   | 0   | 1   | 1   | 1   | 1   | 0   | 0 |
| 168  | 169   | 173 | 172 | 188 | 189 | 185 | 184 | 248 | 249 | 253 | 252 | 236 | 237 | 233 | 232 |   |
| $\overline{e}\overline{f}g\overline{h}$            | 0   | 0   | 1   | 1   | 1   | 0   | 0   | 1   | 0   | 0   | 1   | 1   | 1   | 1   | 0   | 0 |
| 136  | 137   | 141 | 140 | 156 | 157 | 153 | 152 | 216 | 217 | 221 | 220 | 204 | 205 | 201 | 200 |   |
| $\overline{e}\overline{f}gh$                       | 0   | 0   | 1   | 1   | 1   | 0   | 0   | 1   | 0   | 0   | 1   | 1   | 1   | 1   | 0   | 0 |
| 138  | 139   | 143 | 142 | 158 | 159 | 155 | 154 | 218 | 219 | 223 | 222 | 206 | 207 | 203 | 202 |   |
| $\overline{e}fgh$                                  | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0 |
| 130  | 131   | 135 | 134 | 150 | 151 | 147 | 146 | 210 | 211 | 215 | 214 | 198 | 199 | 195 | 194 |   |
| $\overline{e}fgh$                                  | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0 |
| 128  | 129   | 133 | 132 | 148 | 149 | 145 | 144 | 208 | 209 | 213 | 212 | 196 | 197 | 193 | 192 |   |

# One Hot Encoding

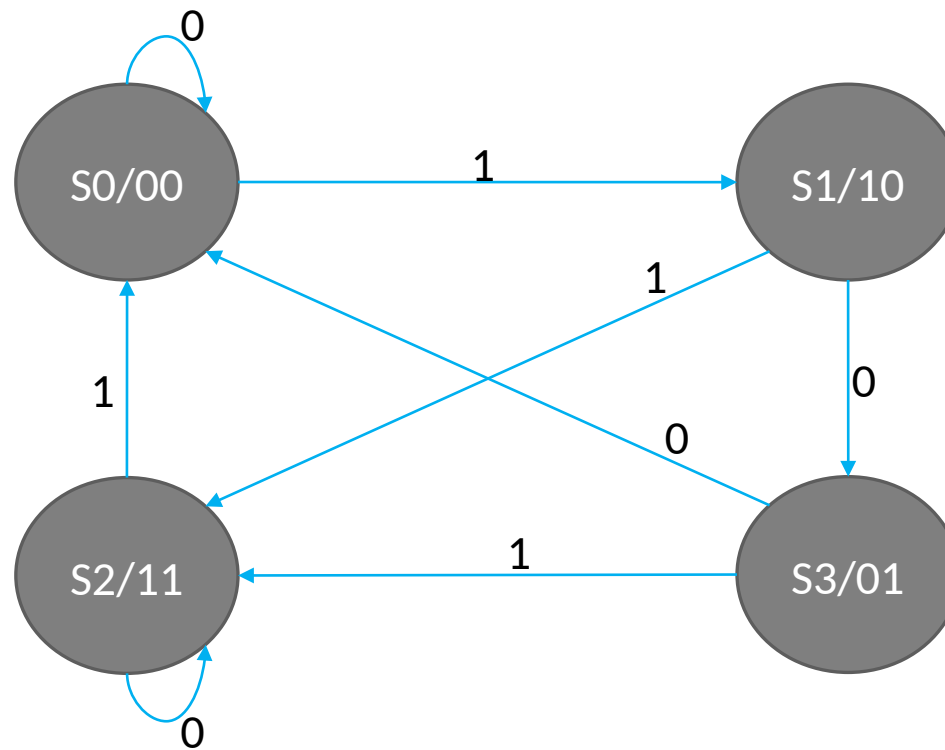
---

- \* Instead of assigning each state a binary number, each state is represented as a single binary value.
- \* Number of bits in the value is the number of states
  - Only one bit can be 1 for each value
  - 1 flip flop for each state
  - One hot encoding uses more flip flops than binary encoding
  - No state table necessary
- \* For 3 states, the encoding is State 0 ->001, State 1 -> 010, State 2 ->100

# Example

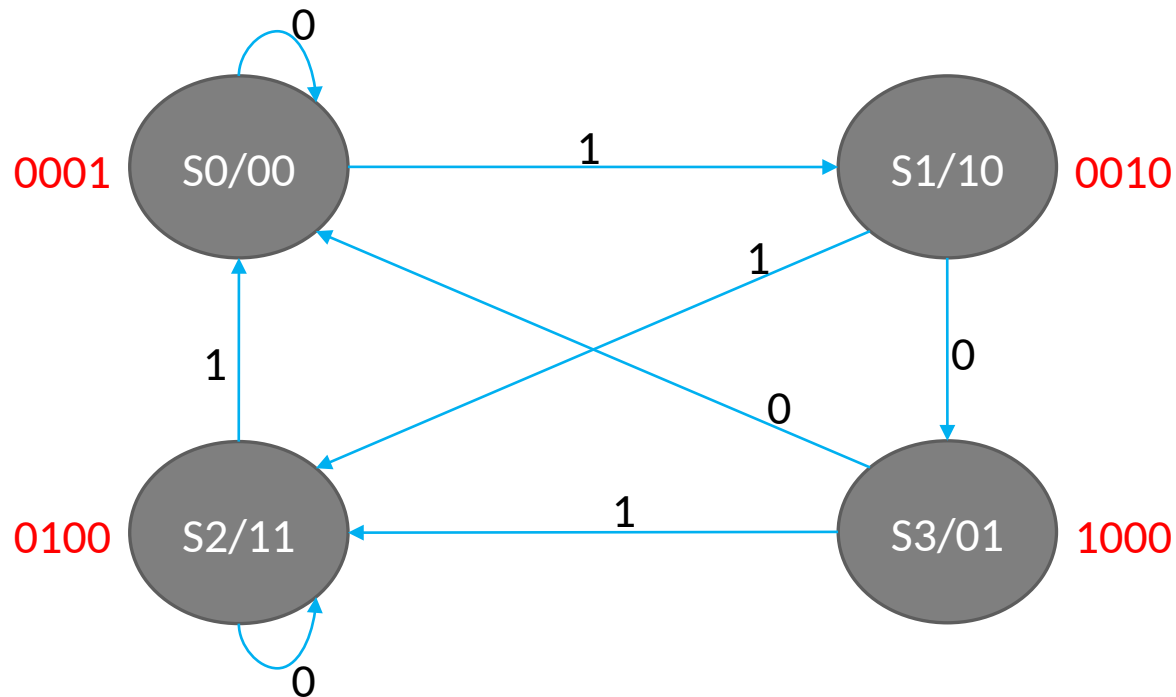
---

Given the state diagram, encode the states with one hot encoding



# Encoding

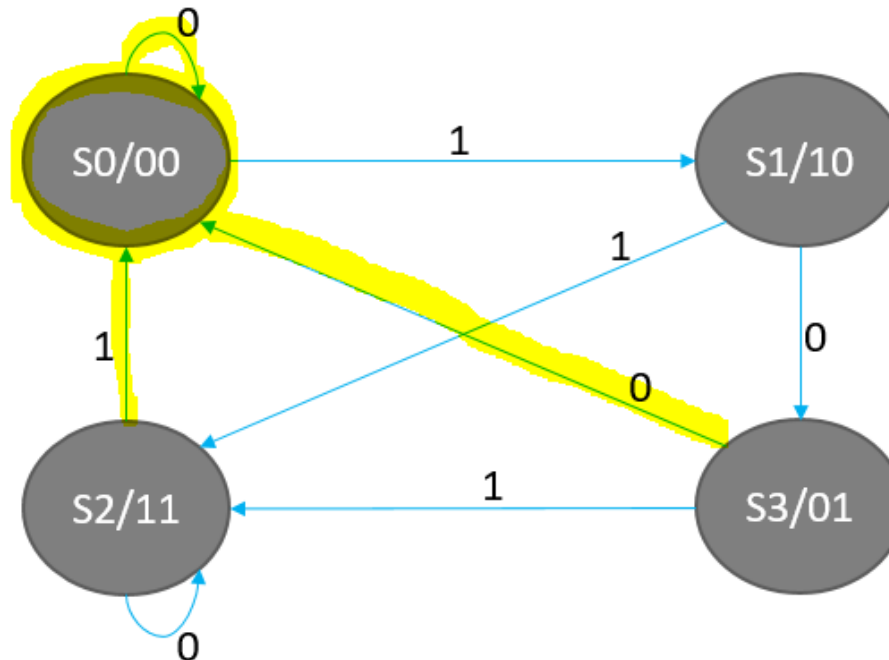
- \* Need 4 bits in encoding because 4 states exist
- \* Each encoded number can only contain a single 1
  - 1 represents output of flip flop. (Will not be using



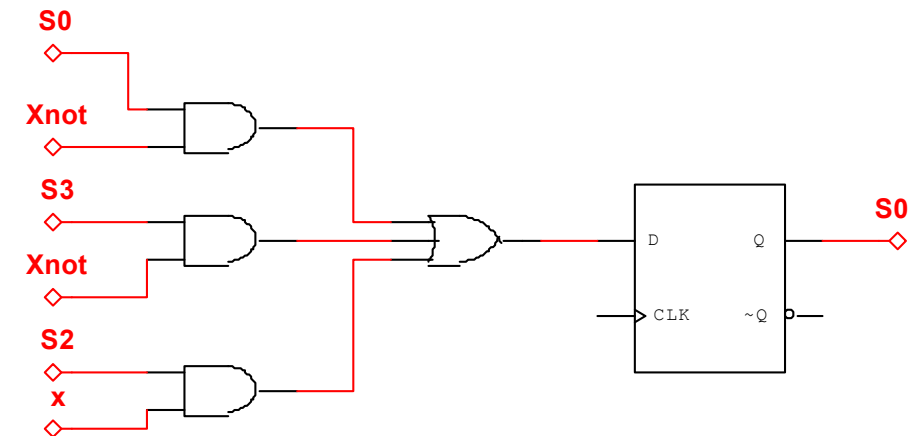
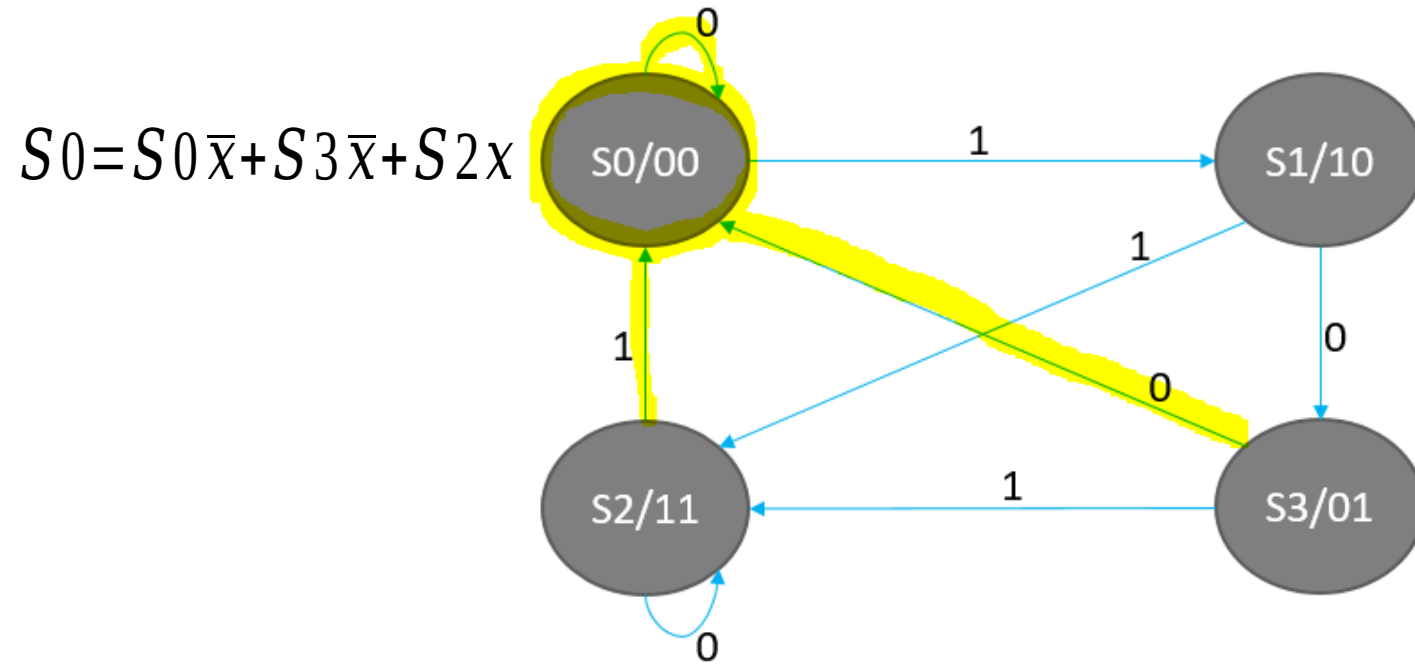
# Next State Logic

- \* Do not need state table
- \* For a given state, look at the transition arrows *into* it, each arrow is an input on an OR gate
  - Connected to the OR gate input is the state coming from AND the input

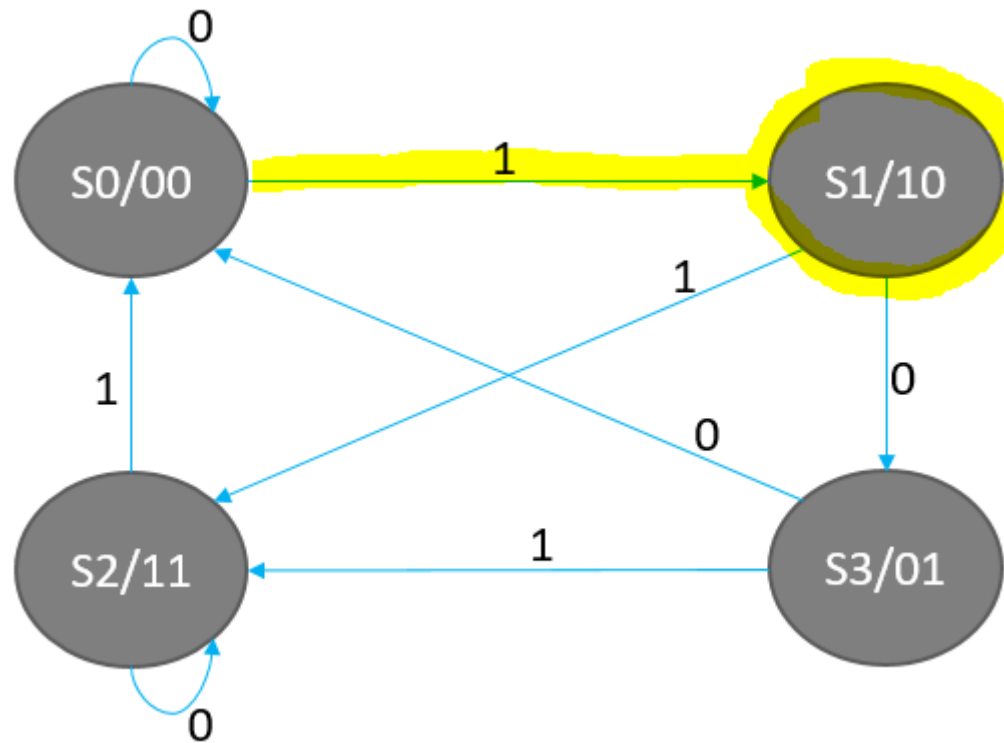
What is the next state logic for S0?



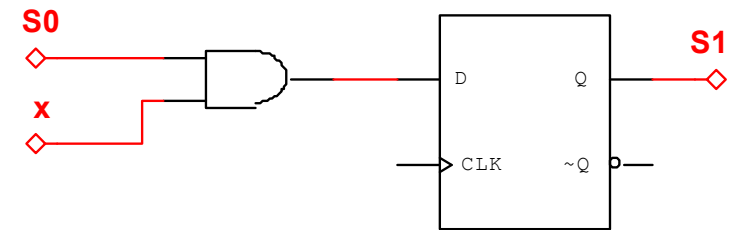
# Next State Logic



# Next State Logic

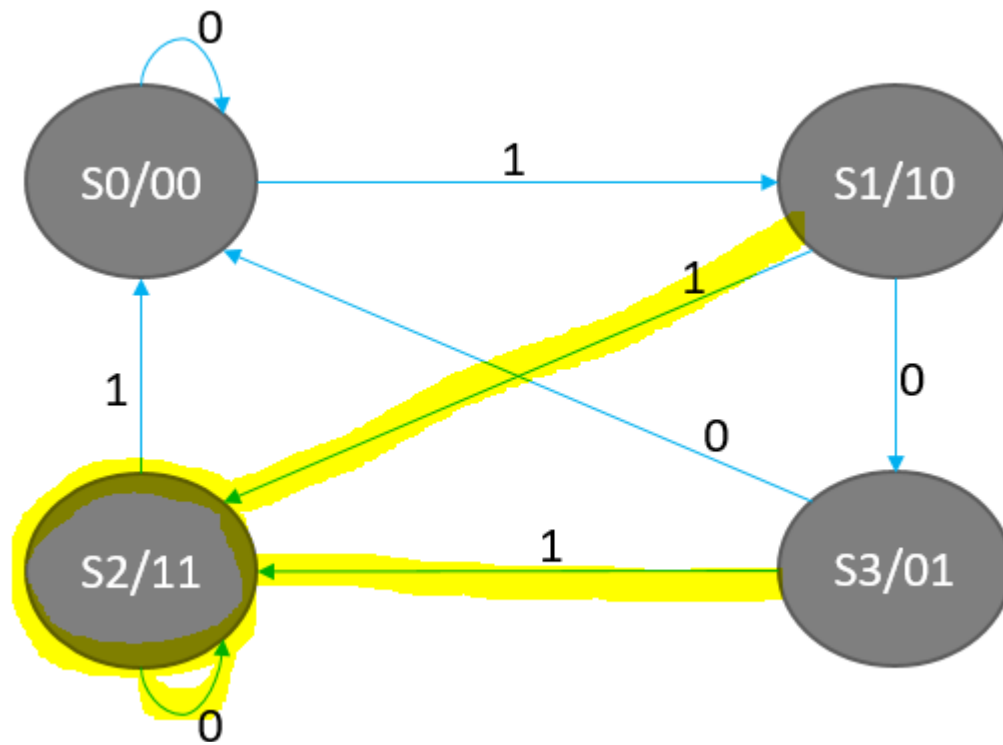


$$S1 = S0x$$

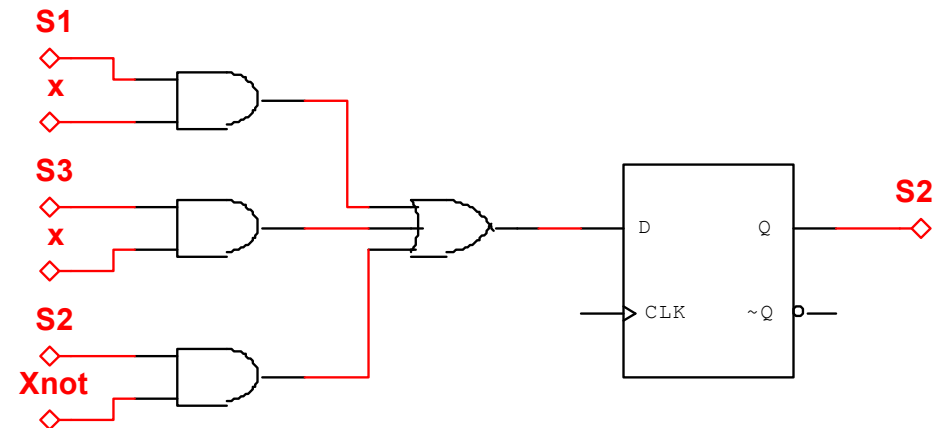




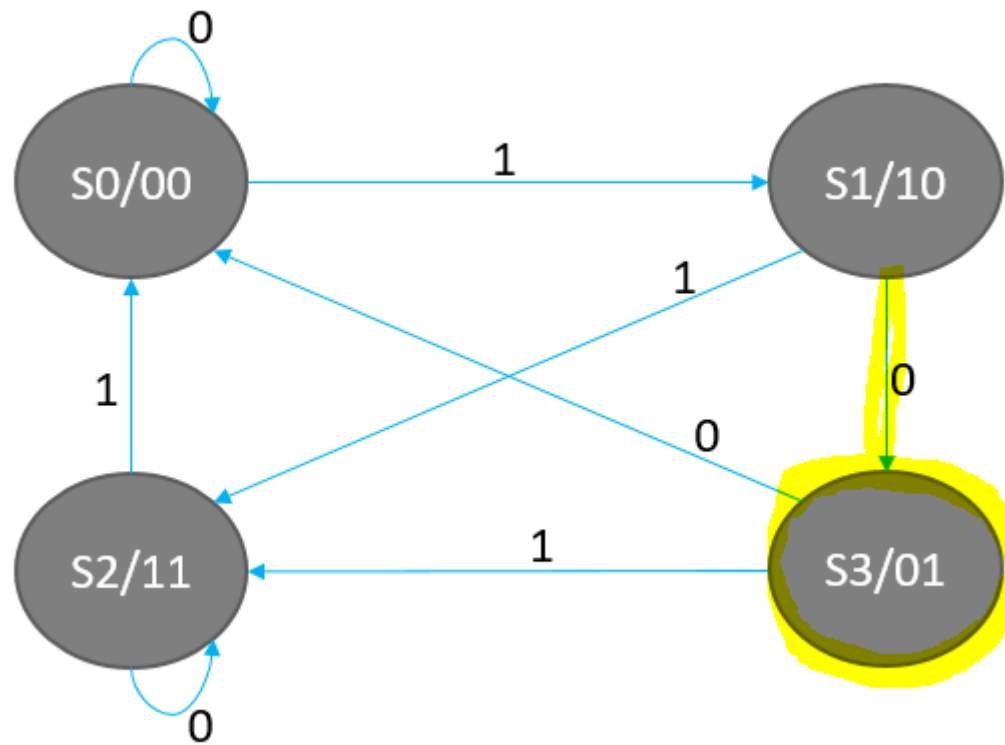
# Next State Logic



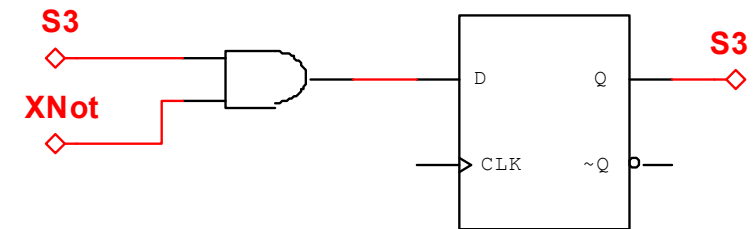
$$S2 = S1x + S3x + S2\bar{x}$$



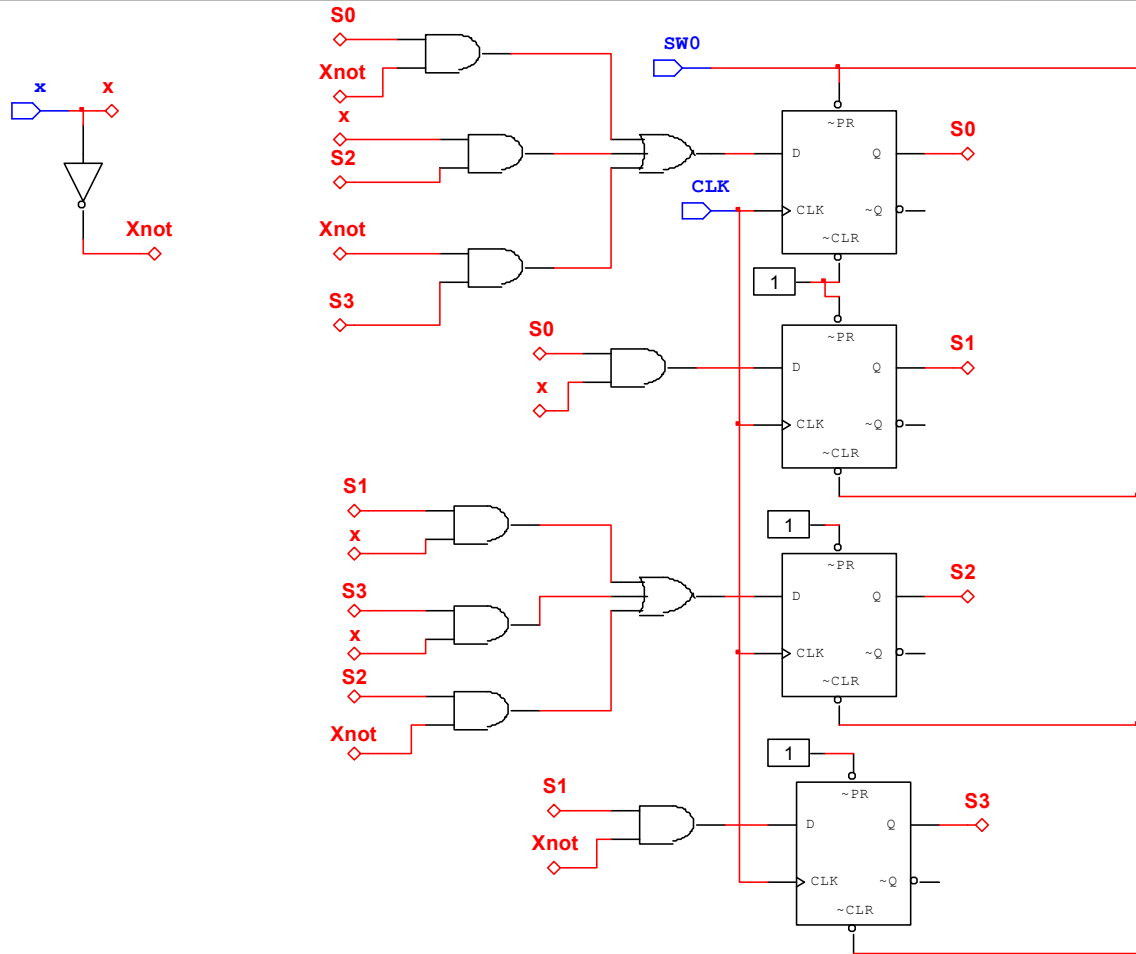
# Next State Logic



$$S3 = S1 \bar{X}$$

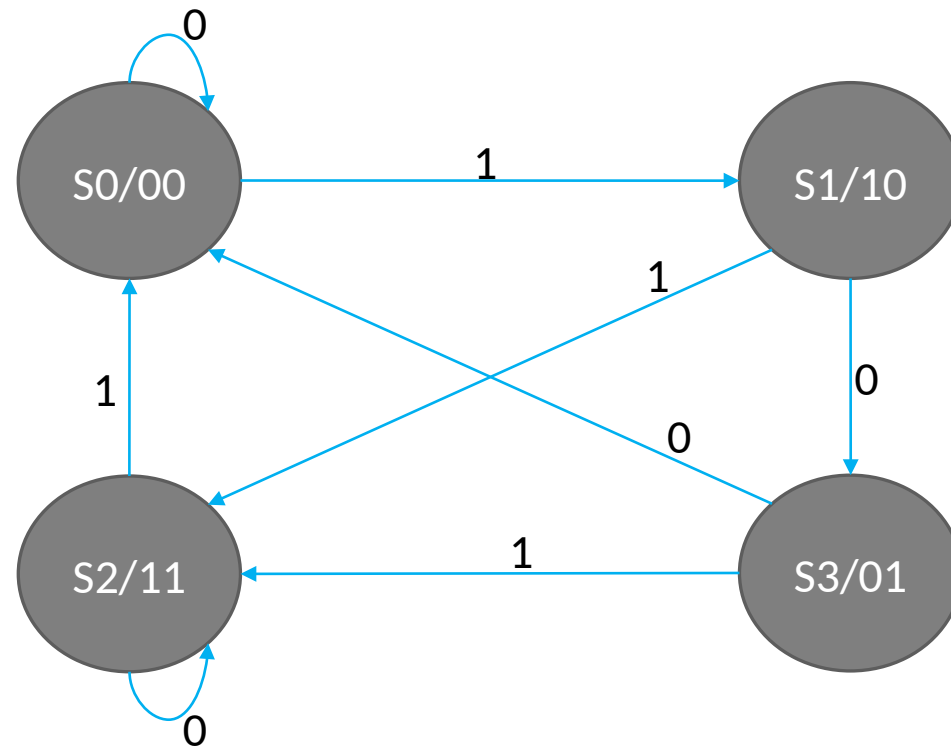


# Next State Logic



# Output Logic

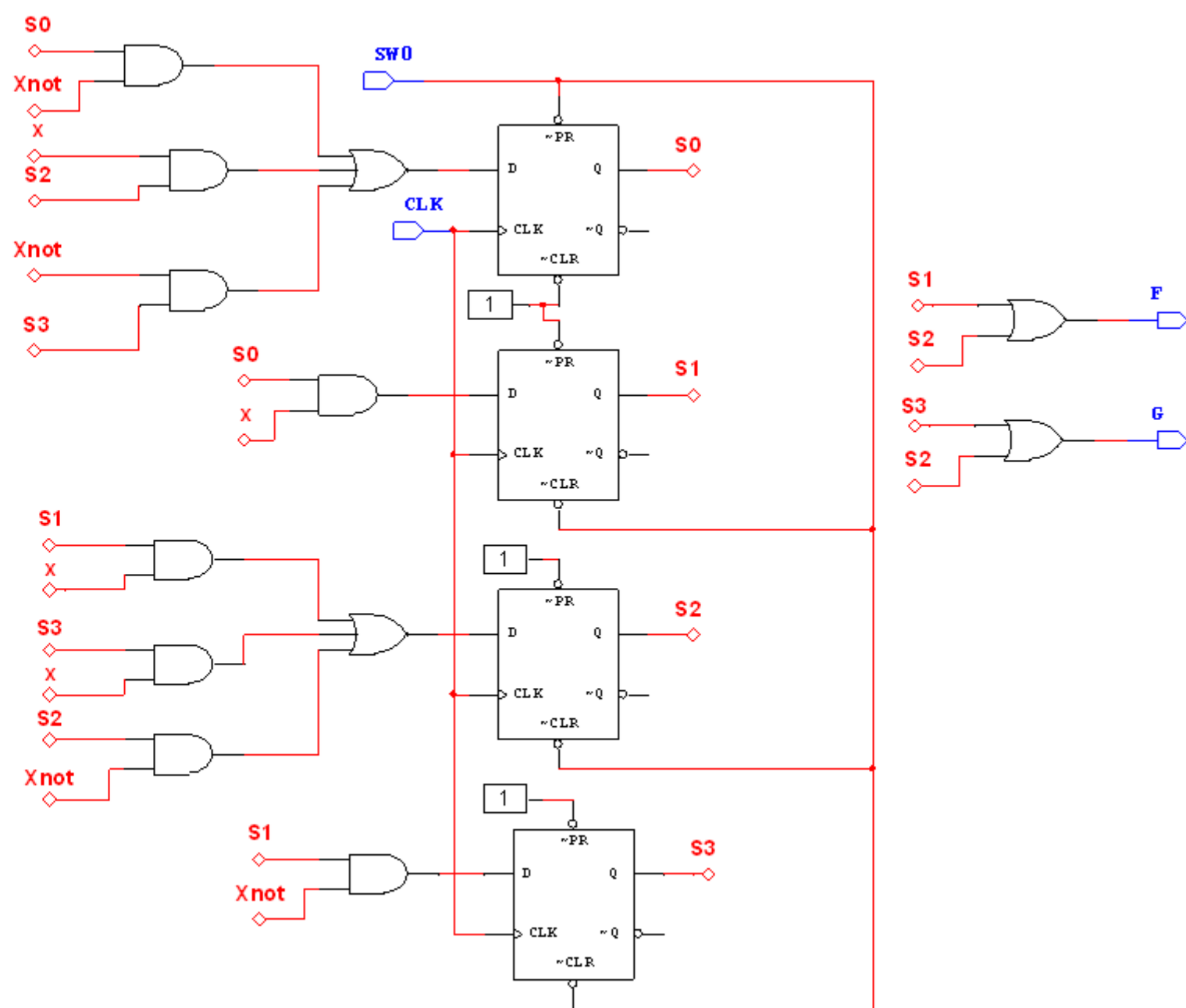
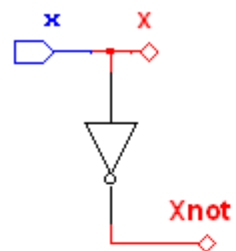
- \* Look at only the states where the output is 1
  - OR all these states together



$$F = S1 + S2$$

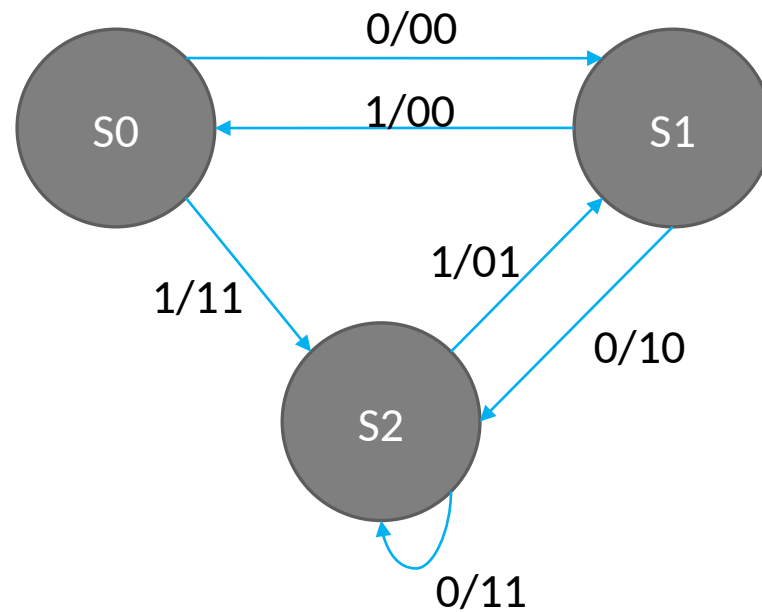
$$G = S2 + S3$$

F is MSB output



# Mealy One Hot

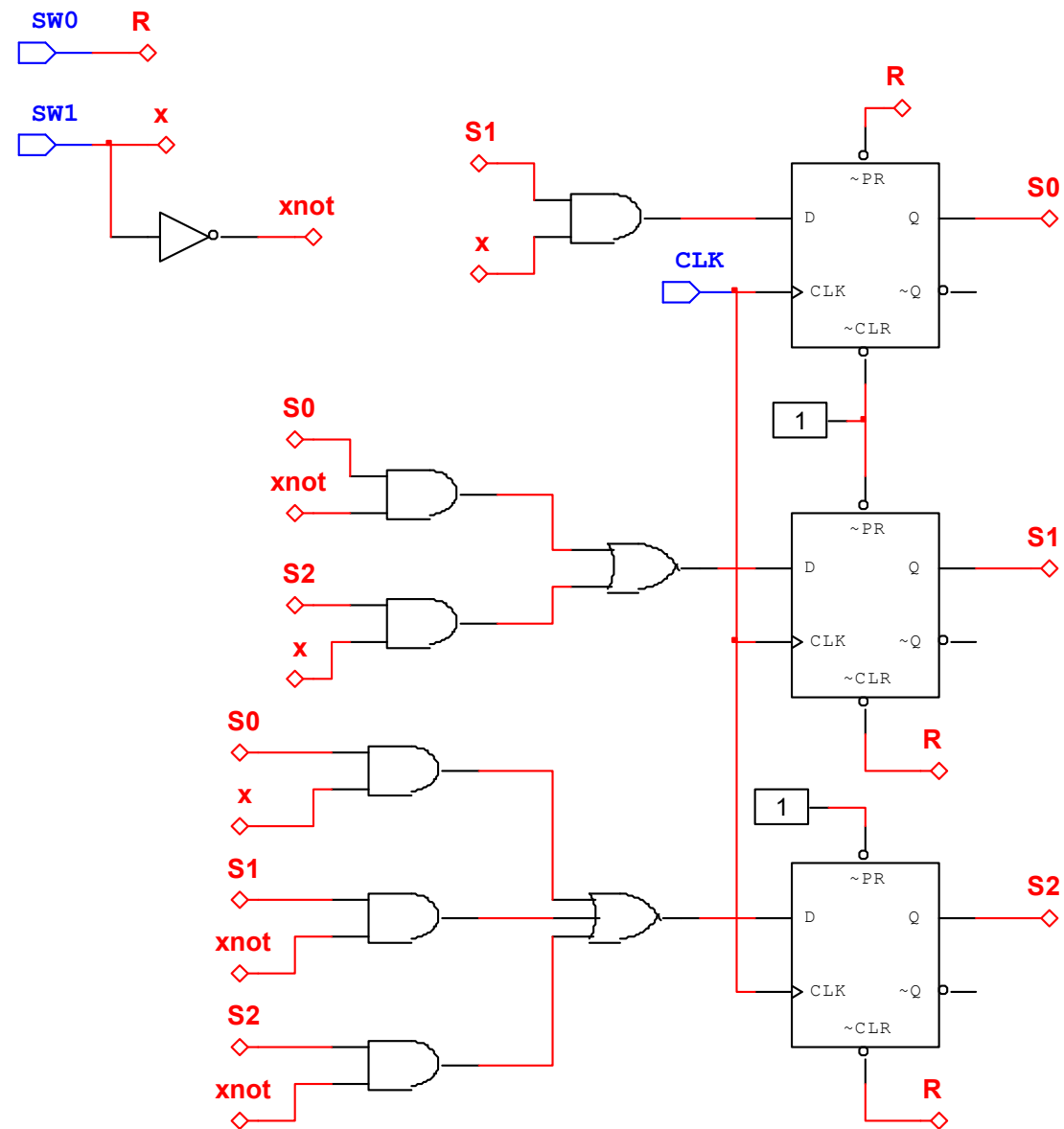
\* Next transition logic is the same approach regardless of if state machine is Moore or Mealy



$$S0 = S1x$$

$$S1 = S0\bar{x} + S2x$$

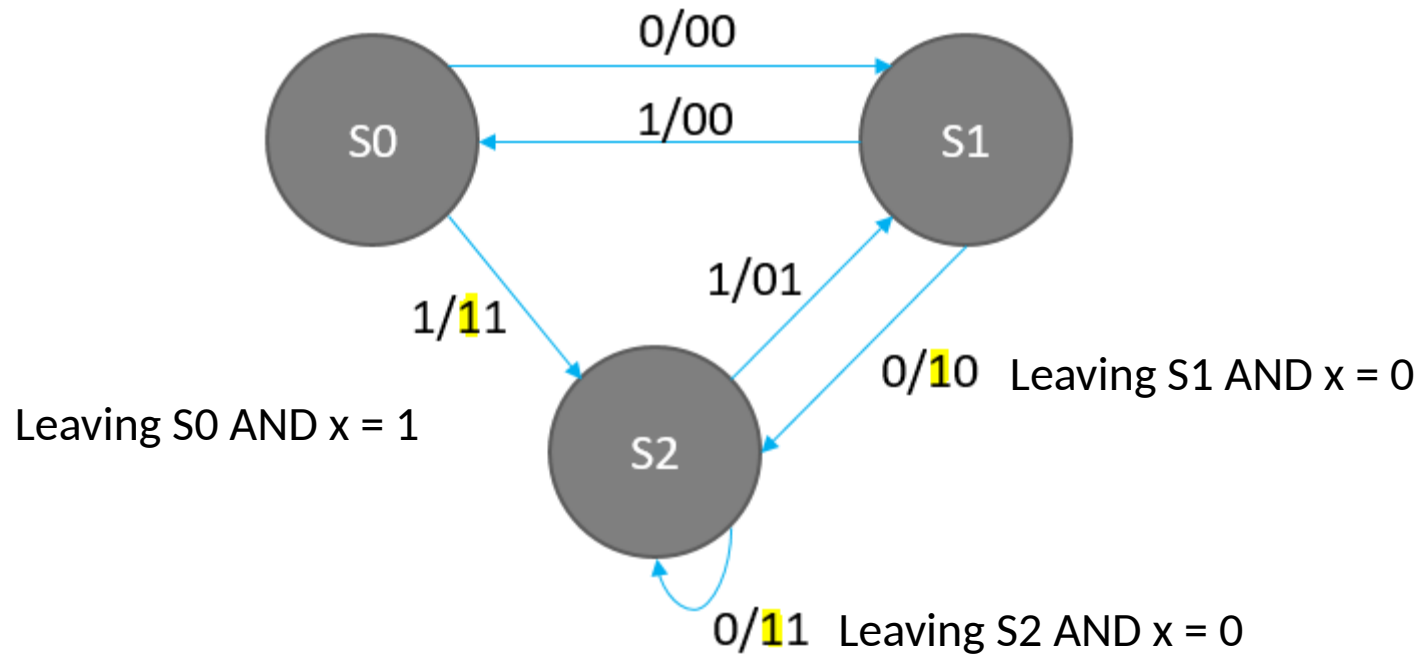
$$S2 = S2\bar{x} + S0x + S1\bar{x}$$



# Mealy Output

\* Find where output is 1 on transition arrow

◦ Output = (state A leaving AND input) + (state B leaving AND input)....

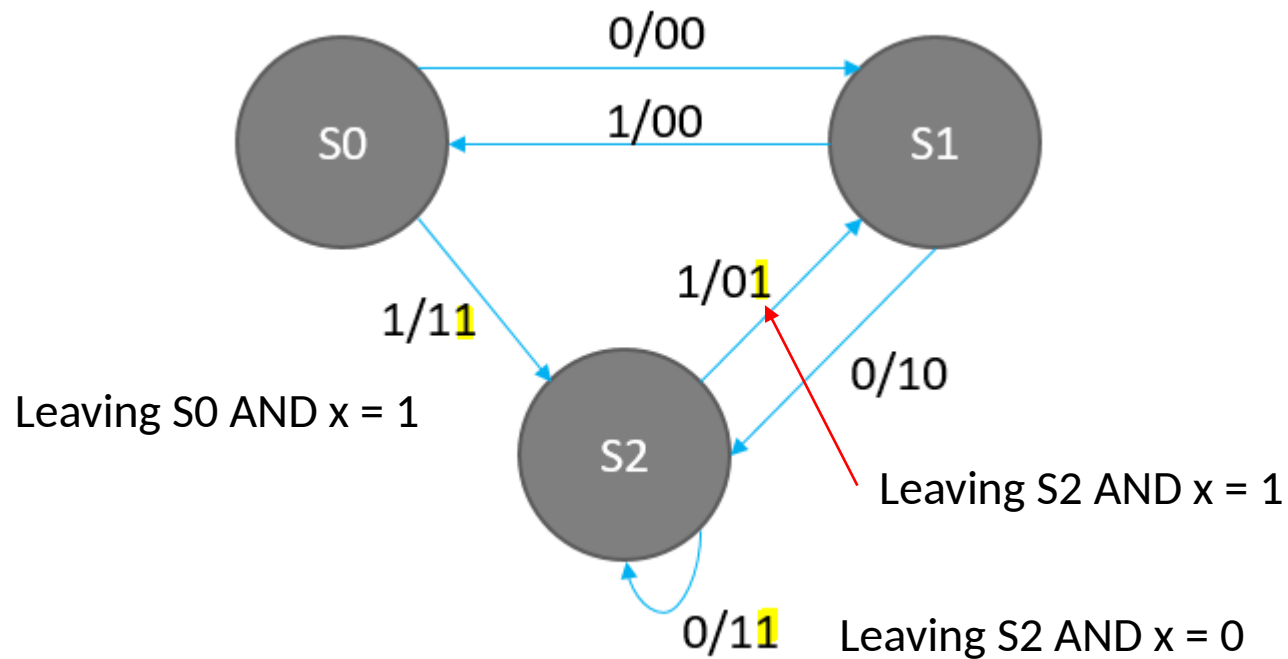


$$F = S_0x + S_1\bar{x} + S_2\bar{x}$$



# Mealy Output

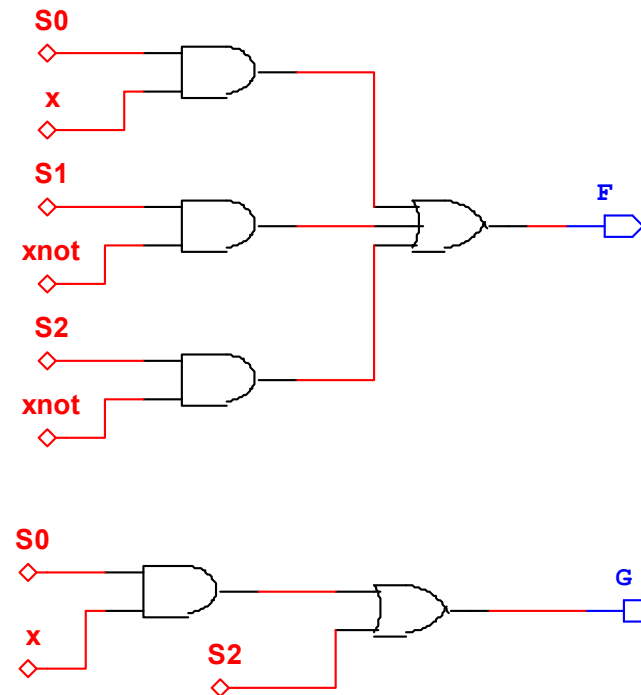
---

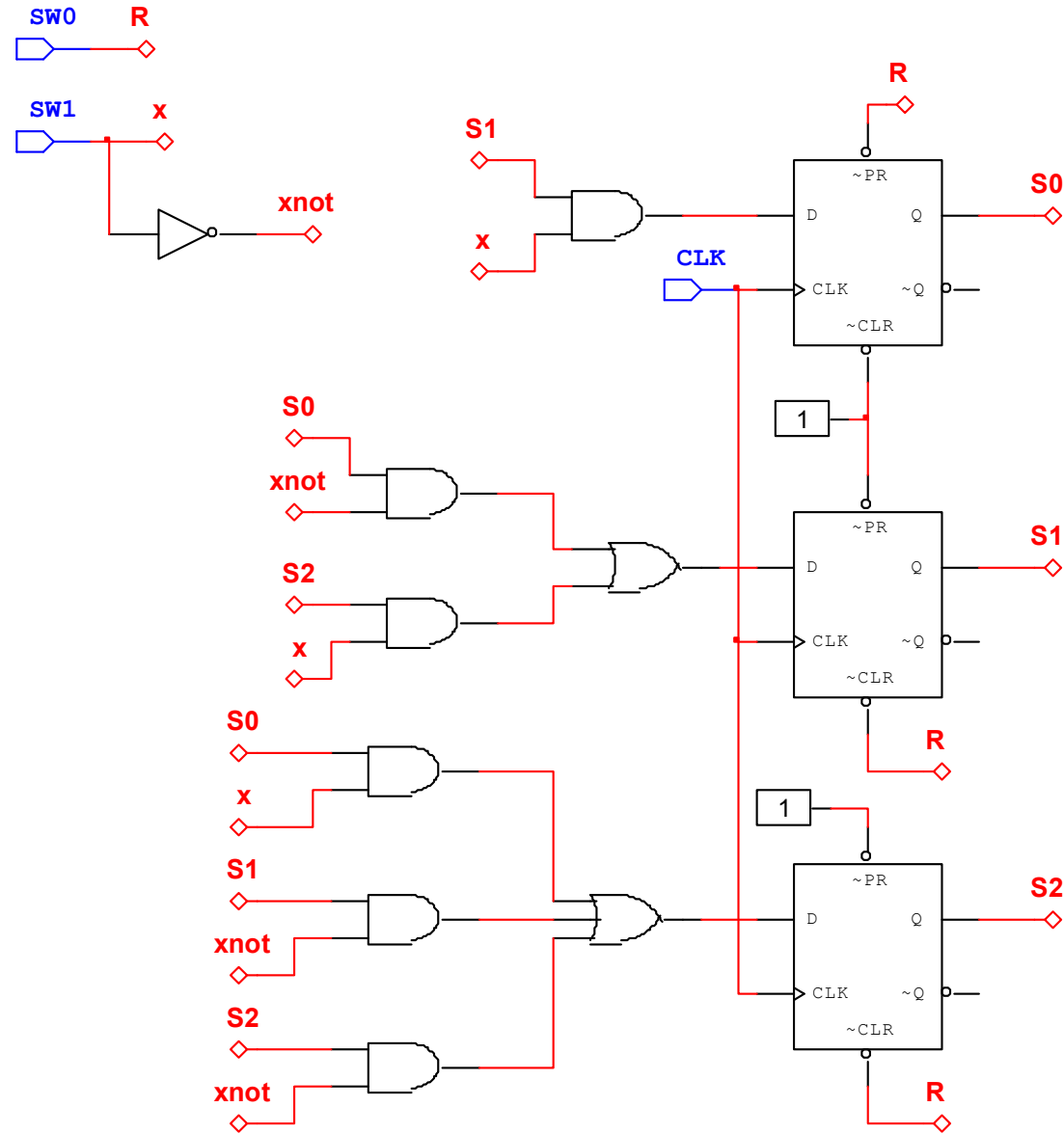


$$G = S_0x + S_2x + S_2\bar{x}$$

# Output One Hot

\* Output logic may be *unsimplified*. Use Boolean algebra to reduce





Why is F not just connected to S2?

