Audrey Ong 3/3/16

CSC 270

Homework 5

Problem 1

Part 1

Figure 1 shows the state diagram of the sequencer that is to be implemented with D flip-flops.

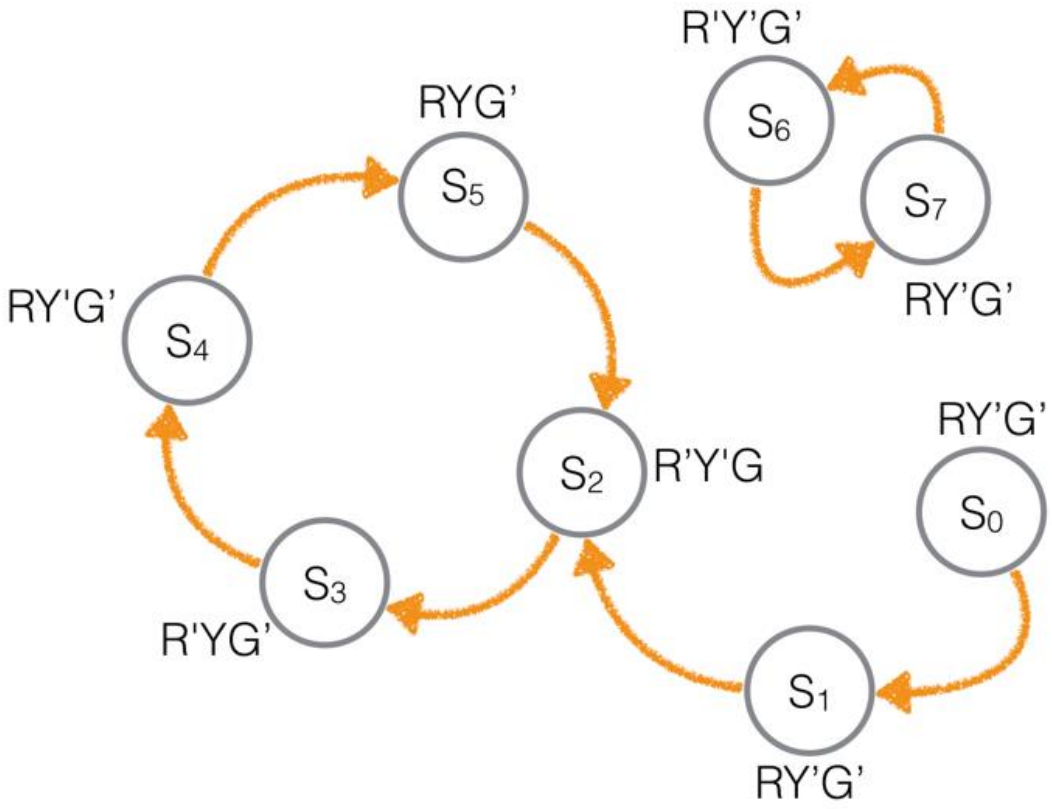

Figure 1. State Diagram of Sequencer to be Implemented

From Figure 1, eight states are observed, which corresponds to three flip-flops. Table 1 shows Figure 1's T_n to T_{n+1} state table, and table 2 shows the binary representation of table 1. The variables S0, S1, S2, S3, S4, S5, S6, and S7 represents the 8 different states observed in the state diagram. The variable Q2, Q1, and Q0 represent individual bits that are the outputs from the three different flip-flops. T_n and T_{n+1} represent the current state and the future state respectively, and D2, D1, and D0 represent the Boolean expression that expresses the future Q2, Q1 and Q0 as a function of the current Q2, Q1 and Q0.

Table 1. T_n to T_{n+1} State Table

T_n	T_{n+1}
S0	S1
S1	S2
S2	S3
S3	S4
S4	S5
S5	S2
S6	S7
S7	S6

Table 2. T_n to T_{n+1} Binary State Table

Present			Future		
T_n			T_{n+1}		
Q2	Q1	Q0	Q2	Q1	Q0
0	0	0	0	0	1
0	0	1	0	1	1
0	1	1	0	1	0
0	1	0	1	1	0
1	1	0	1	1	1
1	1	1	0	1	1
1	0	1	1	0	0
1	0	0	1	0	1
			D2	D1	D0
			T_n		

Tables 3, 4, and 5 show the Karnaugh maps for D2, D1, and D0 respectively, and Equations 1, 2, and 3 show the Boolean expressions for D2, D1, and D0 respectively that come from the Karnaugh map.

Table 3. Karnaugh Map for D2

Q2 \ Q1Q0	00	01	11	10
0				1
1	1	1		1

Table 4. Karnaugh Map for D1

Q2 \ Q1Q0	00	01	11	10
0		1	1	1
1			1	1

Table 5. Karnaugh Map for D0

Q2 \ Q1Q0	00	01	11	10
0	1	1		
1	1		1	1

$$D2 = Q1Q0' + Q2Q1' \quad (Equation\ 1)$$

$$D1 = Q1 + Q2'Q0 \quad (Equation\ 2)$$

$$D0 = Q2'Q1' + Q1'Q0' + Q2Q1 = Q1'(Q2' + Q0') + Q2Q1 \quad (Equation\ 3)$$

Table 6 shows the transition table containing the Q outputs and the RYG outputs.

Table 6. Q2, Q1, Q0 Outputs and its Corresponding R, Y, G Values

	T_n					
State	Q2	Q1	Q0	R	Y	G
S0	0	0	0	1	0	0
S1	0	0	1	1	0	0
S2	0	1	1	0	0	1
S3	0	1	0	0	1	0
S4	1	1	0	1	0	0
S5	1	1	1	1	1	0
S6	1	0	1	0	0	0
S7	1	0	0	1	0	0

Tables 7, 8, and 9 show the Karnaugh maps for R, Y, and G respectively, and Equations 4, 5, and 6 show the Boolean expressions for R, Y, and G respectively that come from the Karnaugh map.

Table 7. Karnaugh Map for R

Q2 \ Q1Q0	00	01	11	10
0	1	1		
1	1		1	1

Table 8. Karnaugh Map for Y

Q2 \ Q1Q0	00	01	11	10
0				1
1			1	

Table 9. Karnaugh Map for G

Q2 \ Q1Q0	00	01	11	10
0			1	
1				

$$R = Q2'Q1' + Q1'Q0' + Q2Q1 = D0 \quad (Equation\ 4)$$

$$Y = Q2'Q1Q0' + Q2Q1Q0 = Q1(Q2'Q0' + Q2Q0) = Q1(Q2 \oplus Q0)' \quad (Equation\ 5)$$

$$G = Q2'Q1Q0 \quad (Equation\ 6)$$

Below shows the python code and its corresponding output used to check the Boolean equations D2, D1, D0, R, Y and G found. The code shows that the equations do indeed produce the state diagram shown in Figure 1, and it that the sequencer works correctly, no matter what state it starts in.

Code:

```
# hw5.py

# Audrey Ong

# homework 5


def NOT( a ):

    return 1 - a


def RYGSequencer(Q2, Q1, Q0, numOfCycles):

    for step in range( numOfCycles ):

        D0 = ( NOT( Q1 ) & ( NOT( Q2 ) | NOT( Q0 ) ) ) | (Q2 & Q1)

        D1 = Q1 | ( NOT(Q2) & Q0 )

        D2 = ( Q1 & NOT( Q0 ) ) | ( Q2 & NOT( Q1 ) )

        R = D0

        Y = Q1 & NOT( Q2 ^ Q0 ) # Q2 ^ Q0 = Q2 xor Q0

        G = NOT( Q2 ) & Q1 & Q0
```

```
        print( "Q2Q1Q0 = %d %d %d | RYG = %d %d %d" % ( Q2, Q1, Q0,
R, Y, G ) )


        Q0 = D0

        Q1 = D1

        Q2 = D2


def main():
    print("Starting test from S0 000")
    RYGSequencer(0, 0, 0, 8)
    print("")
    print("Starting test from S1 001")
    RYGSequencer(0, 0, 1, 7)
    print("")
    print("Starting test from S2 011")
    RYGSequencer(0, 1, 1, 6)
    print("")
    print("Starting test from S3 010")
    RYGSequencer(0, 1, 0, 6)
    print("")
    print("Starting test from S4 110")
    RYGSequencer(1, 1, 0, 6)
    print("")
    print("Starting test from S5 111")
    RYGSequencer(1, 1, 1, 6)
    print("")
    print("Starting test from S6 101")
    RYGSequencer(1, 0, 1, 4)
    print("")
```

```
    print("Starting test from S7 100")
    RYGSequencer(1, 0, 0, 4)

main()
```

Output:

```
Starting test from S0 000
Q2Q1Q0 = 0 0 0 | RYG = 1 0 0
Q2Q1Q0 = 0 0 1 | RYG = 1 0 0
Q2Q1Q0 = 0 1 1 | RYG = 0 0 1
Q2Q1Q0 = 0 1 0 | RYG = 0 1 0
Q2Q1Q0 = 1 1 0 | RYG = 1 0 0
Q2Q1Q0 = 1 1 1 | RYG = 1 1 0
Q2Q1Q0 = 0 1 1 | RYG = 0 0 1
Q2Q1Q0 = 0 1 0 | RYG = 0 1 0

Starting test from S1 001
Q2Q1Q0 = 0 0 1 | RYG = 1 0 0
Q2Q1Q0 = 0 1 1 | RYG = 0 0 1
Q2Q1Q0 = 0 1 0 | RYG = 0 1 0
Q2Q1Q0 = 1 1 0 | RYG = 1 0 0
Q2Q1Q0 = 1 1 1 | RYG = 1 1 0
Q2Q1Q0 = 0 1 1 | RYG = 0 0 1
Q2Q1Q0 = 0 1 0 | RYG = 0 1 0

Starting test from S2 011
Q2Q1Q0 = 0 1 1 | RYG = 0 0 1
Q2Q1Q0 = 0 1 0 | RYG = 0 1 0
Q2Q1Q0 = 1 1 0 | RYG = 1 0 0
```

```
Q2Q1Q0 = 1 1 1 | RYG = 1 1 0

Q2Q1Q0 = 0 1 1 | RYG = 0 0 1

Q2Q1Q0 = 0 1 0 | RYG = 0 1 0


Starting test from S3 010

Q2Q1Q0 = 0 1 0 | RYG = 0 1 0

Q2Q1Q0 = 1 1 0 | RYG = 1 0 0

Q2Q1Q0 = 1 1 1 | RYG = 1 1 0

Q2Q1Q0 = 0 1 1 | RYG = 0 0 1

Q2Q1Q0 = 0 1 0 | RYG = 0 1 0

Q2Q1Q0 = 1 1 0 | RYG = 1 0 0


Starting test from S4 110

Q2Q1Q0 = 1 1 0 | RYG = 1 0 0

Q2Q1Q0 = 1 1 1 | RYG = 1 1 0

Q2Q1Q0 = 0 1 1 | RYG = 0 0 1

Q2Q1Q0 = 0 1 0 | RYG = 0 1 0

Q2Q1Q0 = 1 1 0 | RYG = 1 0 0

Q2Q1Q0 = 1 1 1 | RYG = 1 1 0


Starting test from S5 111

Q2Q1Q0 = 1 1 1 | RYG = 1 1 0

Q2Q1Q0 = 0 1 1 | RYG = 0 0 1

Q2Q1Q0 = 0 1 0 | RYG = 0 1 0

Q2Q1Q0 = 1 1 0 | RYG = 1 0 0

Q2Q1Q0 = 1 1 1 | RYG = 1 1 0

Q2Q1Q0 = 0 1 1 | RYG = 0 0 1


Starting test from S6 101
```

```
Q2Q1Q0 = 1 0 1 | RYG = 0 0 0

Q2Q1Q0 = 1 0 0 | RYG = 1 0 0

Q2Q1Q0 = 1 0 1 | RYG = 0 0 0

Q2Q1Q0 = 1 0 0 | RYG = 1 0 0


Starting test from S7 100

Q2Q1Q0 = 1 0 0 | RYG = 1 0 0

Q2Q1Q0 = 1 0 1 | RYG = 0 0 0

Q2Q1Q0 = 1 0 0 | RYG = 1 0 0

Q2Q1Q0 = 1 0 1 | RYG = 0 0 0
```

Given the equations found for D2, D1, D0, R, Y and G, the circuit diagram showing the implementation of the state diagram in Figure 1 with D flip-flops is shown in Figure 2.

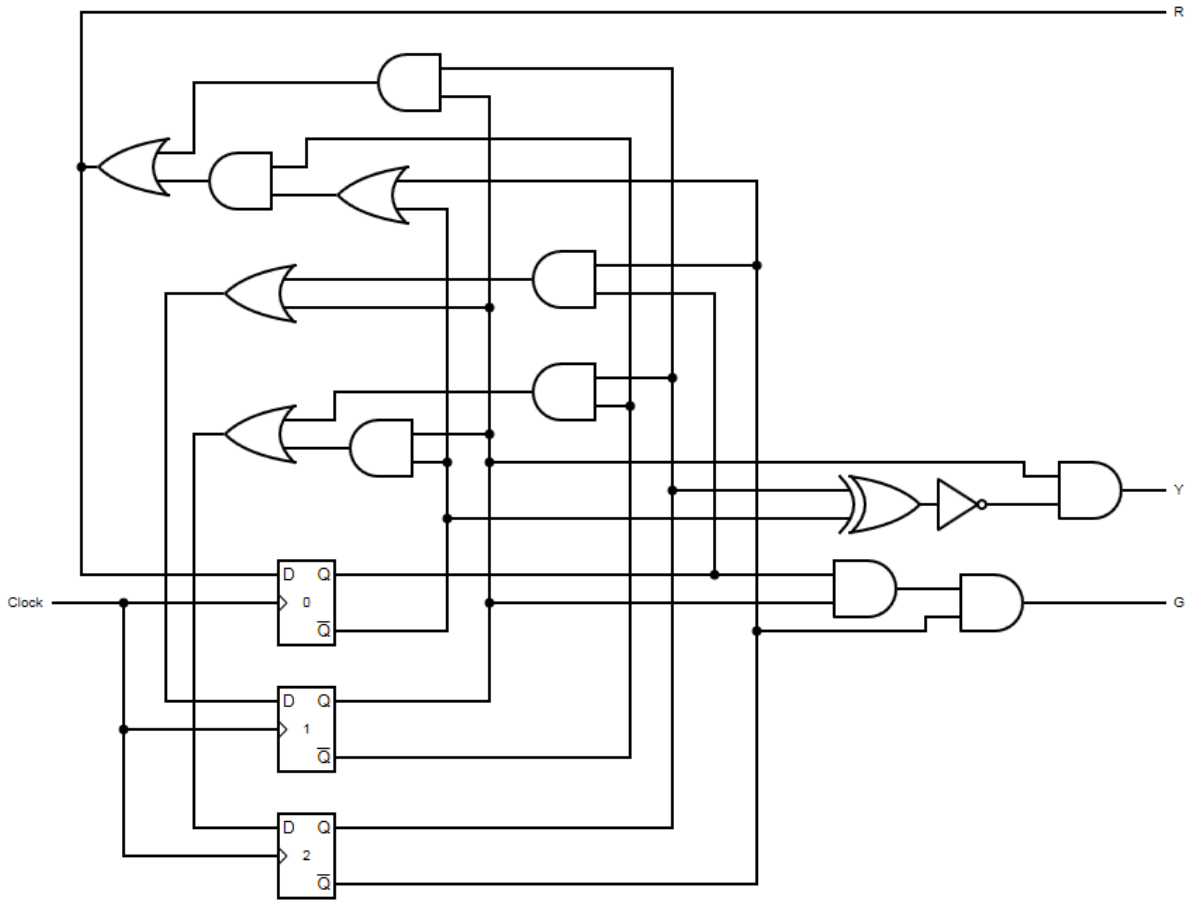

Figure 2. Circuit Diagram of the Sequencer Shown in Figure 1

Part 2

To automatically escape the S6-S7 cycle with the cheapest fix, we need to examine Table 2 and the Karnaugh maps in Tables 3, 4, and 5 and suggest modifications to Table 2 that would require the least amount of change to our circuit diagram shown in Figure 2. As seen from Table 2 and the Karnaugh map in Table 3, if we change Q2 in T_{n+1} from 1 to 0 in both S6 (Q2Q1Q0 = 101 during T_n) and S7 (Q2Q1Q0 = 100 during T_n), we can remove the bottom two 1s in Table 3, which would in fact simply our final circuit, and give us a negative cost. The modified Boolean expression for D2 is expressed in Equation 7.

$$D2(modified) = Q1Q0' \quad (Equation\ 7)$$

Therefore if the sequencer starts in S6, the next state will be S0, and if the sequencer starts in S7 the next state will become S1, thus escaping the S6-S7 loop. Where in Equation 1, D2 originally needed two AND gates and one OR gate to implement, the modified D2 equation only requires one AND gate. Since there are four AND gates in a chip, and four OR gates in a chip, the cost of my fix is -1/4 - 1/4 = -1/2. Figure 3 shows this modified circuit that automatically exits the S6-S7 cycle.

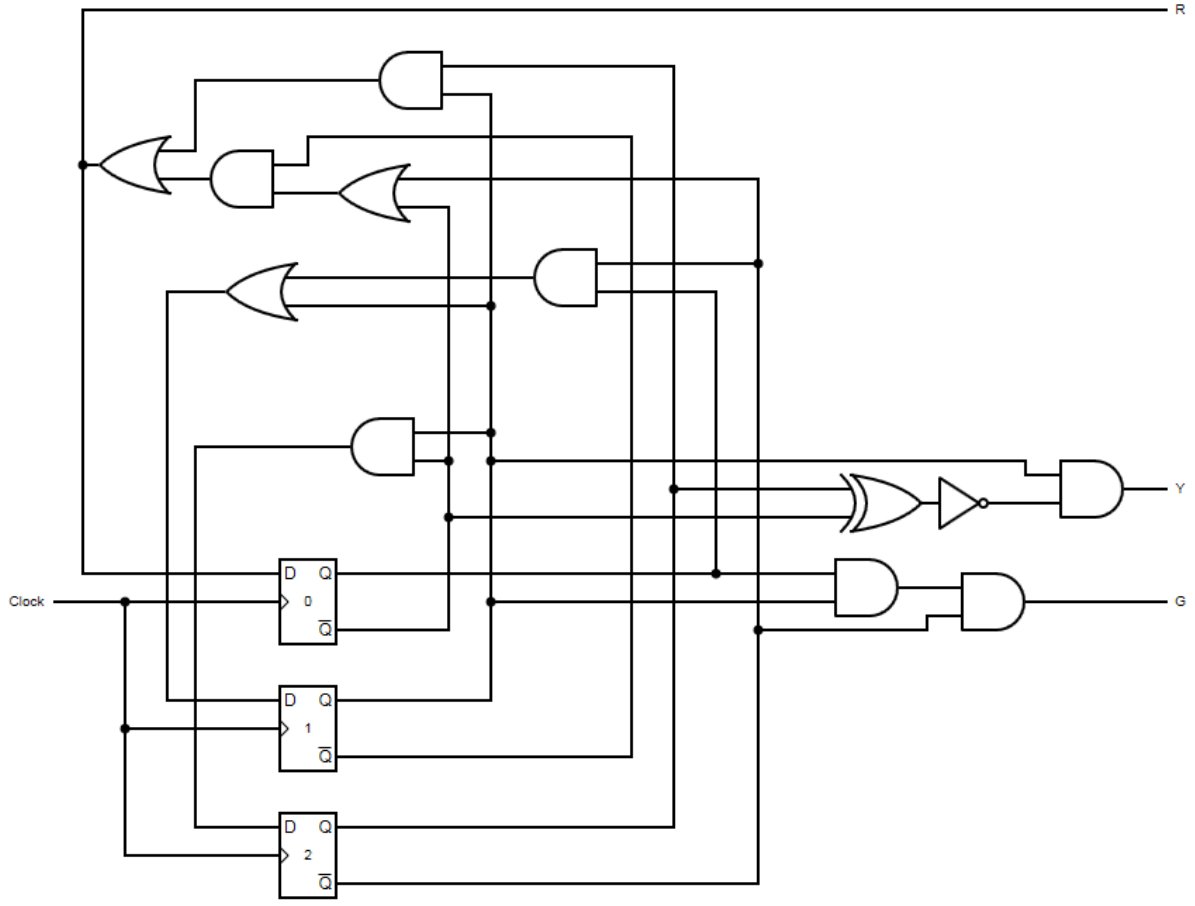

Figure 3. Modified Circuit Diagram of Sequencer that Escapes the S6-S7 Cycle

Below shows the python code used to test the modified circuit that eliminates the S6-S7 cycle. As seen from the code output, the new circuit does indeed escape the S6-S7 cycle automatically.

Code 2:

```
# hw5mod.py

# Audrey Ong

# homework 5


def NOT( a ):

    return 1 - a


def RYGSequencer(Q2, Q1, Q0, numOfCycles):

    for step in range( numOfCycles ):

        D0 = ( NOT( Q1 ) & ( NOT( Q2 ) | NOT( Q0 ) ) ) | (Q2 & Q1)

        D1 = Q1 | ( NOT(Q2) & Q0 )

        D2 = ( Q1 & NOT( Q0 ) )

        R = D0

        Y = Q1 & NOT( Q2 ^ Q0 ) # Q2 ^ Q0 = Q2 xor Q0

        G = NOT( Q2 ) & Q1 & Q0


        print( "Q2Q1Q0 = %d %d %d | RYG = %d %d %d" % ( Q2, Q1, Q0,
R, Y, G ) )


        Q0 = D0

        Q1 = D1

        Q2 = D2


def main():

    print("Starting test from S0 000")

    RYGSequencer(0, 0, 0, 8)
```

```
    print("")
    print("Starting test from S1 001")
    RYGSequencer(0, 0, 1, 7)
    print("")
    print("Starting test from S2 011")
    RYGSequencer(0, 1, 1, 6)
    print("")
    print("Starting test from S3 010")
    RYGSequencer(0, 1, 0, 6)
    print("")
    print("Starting test from S4 110")
    RYGSequencer(1, 1, 0, 6)
    print("")
    print("Starting test from S5 111")
    RYGSequencer(1, 1, 1, 6)
    print("")
    print("Starting test from S6 101")
    RYGSequencer(1, 0, 1, 7)
    print("")
    print("Starting test from S7 100")
    RYGSequencer(1, 0, 0, 7)

main()
```

Output 2:

```
Starting test from S0 000
Q2Q1Q0 = 0 0 0 | RYG = 1 0 0
Q2Q1Q0 = 0 0 1 | RYG = 1 0 0
Q2Q1Q0 = 0 1 1 | RYG = 0 0 1
```

```
Q2Q1Q0 = 0 1 0 | RYG = 0 1 0
Q2Q1Q0 = 1 1 0 | RYG = 1 0 0
Q2Q1Q0 = 1 1 1 | RYG = 1 1 0
Q2Q1Q0 = 0 1 1 | RYG = 0 0 1
Q2Q1Q0 = 0 1 0 | RYG = 0 1 0

Starting test from S1 001
Q2Q1Q0 = 0 0 1 | RYG = 1 0 0
Q2Q1Q0 = 0 1 1 | RYG = 0 0 1
Q2Q1Q0 = 0 1 0 | RYG = 0 1 0
Q2Q1Q0 = 1 1 0 | RYG = 1 0 0
Q2Q1Q0 = 1 1 1 | RYG = 1 1 0
Q2Q1Q0 = 0 1 1 | RYG = 0 0 1
Q2Q1Q0 = 0 1 0 | RYG = 0 1 0

Starting test from S2 011
Q2Q1Q0 = 0 1 1 | RYG = 0 0 1
Q2Q1Q0 = 0 1 0 | RYG = 0 1 0
Q2Q1Q0 = 1 1 0 | RYG = 1 0 0
Q2Q1Q0 = 1 1 1 | RYG = 1 1 0
Q2Q1Q0 = 0 1 1 | RYG = 0 0 1
Q2Q1Q0 = 0 1 0 | RYG = 0 1 0

Starting test from S3 010
Q2Q1Q0 = 0 1 0 | RYG = 0 1 0
Q2Q1Q0 = 1 1 0 | RYG = 1 0 0
Q2Q1Q0 = 1 1 1 | RYG = 1 1 0
Q2Q1Q0 = 0 1 1 | RYG = 0 0 1
Q2Q1Q0 = 0 1 0 | RYG = 0 1 0
```

```
Q2Q1Q0 = 1 1 0 | RYG = 1 0 0

Starting test from S4 110
Q2Q1Q0 = 1 1 0 | RYG = 1 0 0
Q2Q1Q0 = 1 1 1 | RYG = 1 1 0
Q2Q1Q0 = 0 1 1 | RYG = 0 0 1
Q2Q1Q0 = 0 1 0 | RYG = 0 1 0
Q2Q1Q0 = 1 1 0 | RYG = 1 0 0
Q2Q1Q0 = 1 1 1 | RYG = 1 1 0

Starting test from S5 111
Q2Q1Q0 = 1 1 1 | RYG = 1 1 0
Q2Q1Q0 = 0 1 1 | RYG = 0 0 1
Q2Q1Q0 = 0 1 0 | RYG = 0 1 0
Q2Q1Q0 = 1 1 0 | RYG = 1 0 0
Q2Q1Q0 = 1 1 1 | RYG = 1 1 0
Q2Q1Q0 = 0 1 1 | RYG = 0 0 1

Starting test from S6 101
Q2Q1Q0 = 1 0 1 | RYG = 0 0 0
Q2Q1Q0 = 0 0 0 | RYG = 1 0 0
Q2Q1Q0 = 0 0 1 | RYG = 1 0 0
Q2Q1Q0 = 0 1 1 | RYG = 0 0 1
Q2Q1Q0 = 0 1 0 | RYG = 0 1 0
Q2Q1Q0 = 1 1 0 | RYG = 1 0 0
Q2Q1Q0 = 1 1 1 | RYG = 1 1 0

Starting test from S7 100
Q2Q1Q0 = 1 0 0 | RYG = 1 0 0
```

Q2Q1Q0 = 0 0 1 | RYG = 1 0 0

Q2Q1Q0 = 0 1 1 | RYG = 0 0 1

Q2Q1Q0 = 0 1 0 | RYG = 0 1 0

Q2Q1Q0 = 1 1 0 | RYG = 1 0 0

Q2Q1Q0 = 1 1 1 | RYG = 1 1 0

Q2Q1Q0 = 0 1 1 | RYG = 0 0 1