Tables to be used during Lab 5 development

During in this lab, we consider using Synch. Sequential circuit to build the car's taillight logical. There are 3 different ways: turn left/right, emergency(4-way-on) and brake. We have think about how does the light work and what happen if they turn on at same time and which one has the higher command. Simply, we should figure out the logical turn left or right in first two part, and then consider how does emergency work. At last think about the brake part.

This table shows left turn:

Turn <Left> (Abstract) Sequence			
Current State	**Current Light Pattern**	**Next State**	
S1	**XXX**	**S2**	
S2	**XXO**	**S3**	
S3	**XOO**	**S4**	
S4	**OOO**	**S1**	

This table shows Right turn

Turn <Right> (Abstract) Sequence			
Current State	**Current Light Pattern**	**Next State**	
S1	**XXX**	**S2**	
S2	**OXX**	**S3**	
S3	**OOX**	**S4**	
S4	**OOO**	**S1**	

This table show left turn address:

<Left> Encoding Table				
<current state>	**Binary**			
	b_2	**b_1**	**b_0**	
S1	**0**	**0**	**0**	
S2	**0**	**0**	**1**	
S3	**0**	**1**	**1**	
S4	**1**	**1**	**1**	

This table show right turn address:

<Right> Encoding Table				
<current state>	Binary			
	b_2	b_1	b_0	
S1	0	0	0	
S2	1	0	0	
S3	1	1	0	
S4	1	1	1	

Organize precious two table together for each turn:

	Implementation State Encoding Table(left)						
State	Binary Encodings of Stored Info (in the order they are stored)					Stored Encoding (binary)	Stored Encoding (hex)
	Current State Binary	Next Address	Current & Next	Current & Next (hex)	Current State (hex)		
S2	000	001	000001	01	0	001	1
S3	001	011	001011	0B	1	011	3
S4	011	111	011111	1F	3	111	7
S1	111	000	111000	38	7	000	0

	Implementation State Encoding Table(right)						
State	Binary Encodings of Stored Info (in the order they are stored)					Stored Encoding (binary)	Stored Encoding (hex)
	Current State Binary	Next Address	Current & Next	Current & Next (hex)	Current State (hex)		
S2	000	100	000100	04	0	100	4
S3	100	110	100110	26	4	110	6
S4	110	111	110111	37	6	111	7
S1	111	000	111000	38	7	000	0

This is brake case:

	Implementation State Encoding Table(Brake)						
State	Binary Encodings of Stored Info (in the order they are stored)					Stored Encoding (binary)	Stored Encoding (hex)
	Current State Binary	Next Address	Current & Next	Current & Next (hex)	Current State (hex)		
S2	000	111	000111	07	7	111	0
S1	111	000	111000	38	0	000	7

Last part(how does the circuit work)

In our final circuit, the system works well as we need. There are 8 cases for each light. (When no turn, brake-off, 4-way-off) nothing goes into the circuit. (When turn, brake-off, 4-way-off) or (When turn, brake-on, 4-way-off) the system only shows turning light like 001 011 111 000(right) or 100 110 111(left), in other words, the turn command higher than brake. (only brake-on, 4-way-on and brake-on or all on) in those three cases would state constant 7(all light up) which only shows brake-on, because emergency(4-way-on) command higher than turning and brake command higher than emergency. At last, there still have two cases, 4-way-on only and 4-way-on & turn left/right. In this situation, light would work for 4-way-on. As I said before, 4-way-on has higher command than turn.

Bonus

In this part, we have to replace any gates to LUT. It looks like we do not have any gate before output, so I decide to switch the wire to add another ROM. As the circuit we can see, the number in the first ROM represent position of each time (1234), the second ROM shows the data in each position (0137). Which make the circle more clear to understand.