ECE2072 Assignment

Avvienash Jaganathan (32281013)

Question 4 Answers

Question 4

Perform a small research and explain how this design may be implemented, adapted, or applied into the real-world scenario. You must include references or citations to justify your explanations. The more references or citations that are included that are relevant, the stronger your explanation and justification will be.

The most obvious use of this sequence counter circuit is to be use as a time base counter such as stopwatches and timers. The iSkip and iRev buttons can be modified to suit better command like “pause”. Further, the concept of these circuit counters can be used to replace analogue clocks to digital clocks which is generally easier to read [1].

These timers can also be integrated to more complex machines that perform various functions base on time. Examples of these machines are like Ovens and Washing machines [1][2][3]. In this case, iSkip could be used to skip an operation and iRev can be used to go back on step. An example of the sequence of operations in washing machine could be like; START then FILL WATER then WASH then DRAIN then FILL WATER then RINSE then DRAIN then OFF [2].

Another common application of these FSM and Counter is Digital Signal Processing (DSP) applications [4]. This specialised micro processing chip can be used to proses a sequence of signals and manipulate it mathematically [5]. Example of the sequences of signals that can be processed may be like voice, audio, video, temperature, and pressure. The 2 buttons can be used to perform any manipulation to the input sequence. Moreover, the circuit could be designed as a programable DSP. This gives the users the flexibility to design the DSP according to their needs and preference.

Another real-life application of this design is traffic lights or even smart traffic lights [6]. Each traffic light has a sequence of three elements, Green, Yellow and Red. Whether it holds its output or proceeds to next state depends on the presence of cars or road crossers as well as current states of the other traffic lights at the same intersection. Based on the inputs as well as present state, the next state will be update at a certain point in the clock cycle. This general concept is very similar to the circuit designed in this assignment.

References

[1] https://www.watelectronics.com/what-is-a-digital-counter-types-applications/

[2] https://www.multisim.com/content/Jruvf8ZwPn3AQKgpybJKcf/washing-machine-display-panel/

[3] https://online.visual-paradigm.com/diagrams/templates/state-machine-diagram/oven/

[4] https://www.sciencedirect.com/topics/engineering/finite-state-machine

[5] Application of dynamic counter circuits in programmable digital pixel sensor architectures by M.Habibi

[6] Ghazal, Bilal & Khatib, Khaled & Chahine, Khaled & Kherfan, Mohamad. (2016). Smart traffic light control system. 140-145. 10.1109/EECEA.2016.7470780.

Appendices (Working for Previous Parts)

1 Given Sequence

3 2 6 8 8 0 5 4 4 7 4 5 0 1 8

Since There are 15 states, min number of flip flops are 4 as a bit State is needed

2 State Strategies

2.1 Sequential (Normal): 24 Logic Cells

State		a	b	c	d	Output			
0	**A**	0	0	0	0	0	0	1	1
1	**B**	0	0	0	1	0	0	1	0
2	**C**	0	0	1	0	0	1	1	0
3	**D**	0	0	1	1	1	0	0	0
4	**E**	0	1	0	0	1	0	0	0
5	**F**	0	1	0	1	0	0	0	0
6	**G**	0	1	1	0	0	1	0	1
7	**H**	0	1	1	1	0	1	0	0
8	**I**	1	0	0	0	0	1	0	0
9	**J**	1	0	0	1	0	1	1	1
10	**K**	1	0	1	0	0	1	0	0
11	**L**	1	0	1	1	0	1	0	1
12	**M**	1	1	0	0	0	0	0	0
13	**N**	1	1	0	1	0	0	0	1
14	**O**	1	1	1	0	1	0	0	0
	P	1	1	1	1	X	X	X	X

I0:

ab \ cd	00	01	11	10
00	0	0	1	0
01	1	0	0	0
11	0	0	X	1
10	0	0	0	0

F(a,b,c,d) = A’BC’D’ + A’B’CD + ABC

I1:

ab \ cd	00	01	11	10
00	0	0	0	1
01	0	0	1	1
11	0	0	X	0
10	1	1	1	1

F(a,b,c,d) = AB’ + BCD + A’CD’

I2:

ab \ cd	00	01	11	10
00	1	1	0	1
01	0	0	0	0
11	0	0	X	0
10	0	1	0	0

F(a,b,c,d) = A’B’D’+B’C’D

I3:

ab \ cd	00	01	11	10
00	1	0	0	0
01	0	0	0	1
11	0	1	X	0
10	0	1	1	0

F(a,b,c,d) = A’B’C’D’ + A’BCD’ +AD

2.2 Output Method (b) : 29 Logic Cells

State		a	b	c	d	Output			
0	**A**	1	0	0	0	0	0	1	1
1	**B**	1	0	0	1	0	0	1	0
2	**C**	1	1	0	1	0	1	1	0
3	**D**	0	0	1	1	1	0	0	0
4	**E**	1	0	1	0	1	0	0	0
5	**F**	0	0	0	1	0	0	0	0
6	**G**	0	1	0	0	0	1	0	1
7	**H**	0	1	1	1	0	1	0	0
8	**I**	1	1	1	0	0	1	0	0
9	**J**	1	1	0	0	0	1	1	1
10	**K**	1	1	1	1	0	1	0	0
11	**L**	0	1	1	0	0	1	0	1
12	**M**	0	0	1	0	0	0	0	0
13	**N**	0	0	0	0	0	0	0	1
14	**O**	1	0	1	1	1	0	0	0
	P	0	1	0	1	X	X	X	X

I0:

ab \ cd	00	01	11	10
00	0	0	1	0
01	0	X	0	0
11	0	0	0	0
10	0	0	1	1

F(a,b,c,d) = B'CD + AB'C

I1:

ab \ cd	00	01	11	10
00	0	0	0	0
01	0	X	0	0
11	1	1	1	1
10	1	1	1	1

F(a,b,c,d) = B

I2:

ab \ cd	00	01	11	10
00	0	0	0	0
01	0	X	0	0
11	1	1	0	0
10	1	1	0	0

F(a,b,c,d) = AC'

I3:

ab \ cd	00	01	11	10
00	1	0	0	0
01	1	X	0	1
11	1	0	0	0
10	1	0	0	0

F(a,b,c,d) = C'D' + A'BD'

2.3 Gray Code Method: 29 Logic Cells

State		a	b	c	d	Output			
0	**A**	0	0	0	0	0	0	1	1
1	**B**	0	0	0	1	0	0	1	0
2	**C**	0	0	1	1	0	1	1	0
3	**D**	0	0	1	0	1	0	0	0
4	**E**	0	1	1	0	1	0	0	0
5	**F**	0	1	0	0	0	0	0	0
6	**G**	0	1	0	1	0	1	0	1
7	**H**	0	1	1	1	0	1	0	0
8	**I**	1	1	1	1	0	1	0	0
9	**J**	1	1	0	1	0	1	1	1
10	**K**	1	1	0	0	0	1	0	0
11	**L**	1	1	1	0	0	1	0	1
12	**M**	1	0	1	0	0	0	0	0
13	**N**	1	0	1	1	0	0	0	1
14	**O**	1	0	0	1	1	0	0	0
	P	1	0	0	0	X	X	X	X

I0:

ab \ cd	00	01	11	10
00	0	0	0	1
01	0	0	0	1
11	0	0	0	0
10	X	1	0	0

F(a,b,c,d) = AB'C' + A'CD'

I1:

ab \ cd	00	01	11	10
00	0	0	1	0
01	0	1	1	0
11	1	1	1	1
10	X	0	0	0

F(a,b,c,d) = AB+BD+A'CD

I2:

ab \ cd	00	01	11	10
00	1	1	1	0
01	0	0	0	0
11	0	1	0	0
10	X	0	0	0

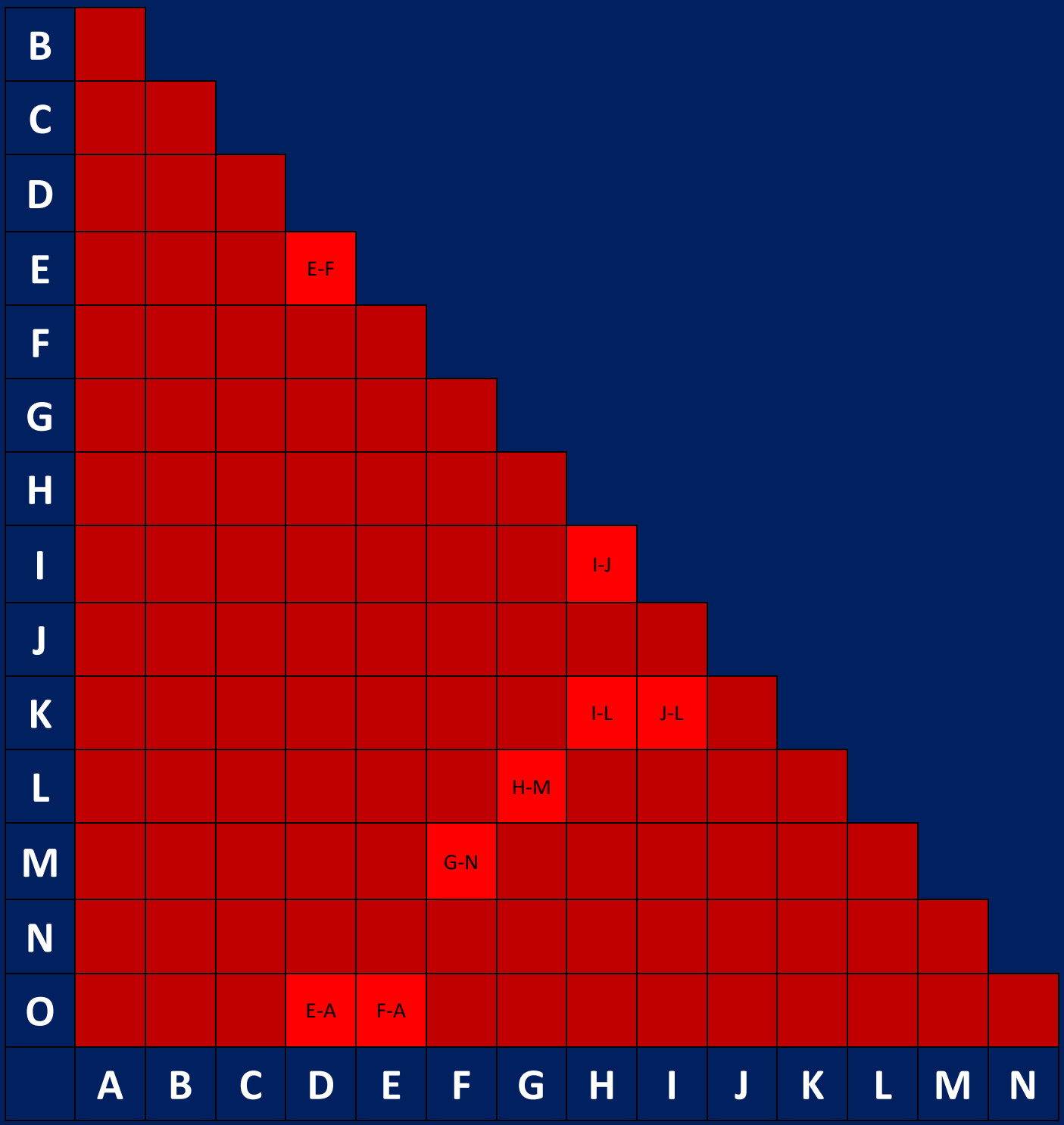
F(a,b,c,d) = ABC'D+A'B'C' +A'B'D

I3:

ab \ cd	00	01	11	10
00	1	0	0	0
01	0	1	0	0
11	0	1	0	1
10	X	0	1	0

F(a,b,c,d) = B'C'D' +BC'D +ABCD'+AB'CD

3 State Minimisation



Therefore, cannot minimise the states

4 Conclusion

The best option is 15 states, standard sequential method.