**Digital Logic Circuit Design Lab**

**Traffic Light Controller FSM**

Combinational Logic Circuit Lab

Name/ID: An Gyeonheal / 21900416

Date: 06-29-2023

# **Problem Introduction**

## Introduction

본 보고서는 간단한 신호등을 FSM으로 구현하는 보고서이다. 신호등은 사거리에서 두 방향의 차량 이동을 고려하였다. 조건에 주어진 상황으로는 B 도로를 기준으로 하여 B 도로에 차가 없는 경우와, B 도로에 차가 있는 경우로 나누어 진행하였다. 첫 번째로, B 도로에 차가 없는 경우에는 A 도로의 신호등은 초록불(G)이 들어온다. 두 번째로, B 도로에 차가 있는 경우에는 A 도로의 신호등은 1초간 노란불이 들어온 후 2초간 빨간불을 유지한다. 그 후 다시 초록불로 2초간 유지한다. 앞서 말한 조건을 수행하기 위해 본 보고서는 Moore FSM을 기준으로 입력과 출력을 정의한 후, 상황을 고려한 State를 결정하였다. 그리고 State graph를 그려 State 간의 상관관계를 도시한 후 State Table을 구성하였고 구성한 State Table의 Present State, Next State와 Output을 Binary로 나타내고 Karnaugh map을 이용하여 minimum SOP를 찾아내었다. 찾아낸 minimum SOP를 통해 Logisim 프로그램을 활용하여 회로를 구성하고 시뮬레이션을 실행하여 정상적으로 FSM이 작동하는지 확인하였다.

# **Design Process**

### Input & Output Definition

**INPUTS**

V={0, 1} – Road B 차의 유무

CLK is 1 sec

**OUTPUTS :**

T={G,Y, R} – Road A의 신호

{1, 0, 0} – 초록불 점등

{0, 1, 0} – 노란불 점등

{0, 0, 1} – 빨간불 점등

### State Description

|  |  |
| --- | --- |
| **States** | **Description** |
|  | Road B에 차량 없음  (T=G for 1 sec) |
|  | Road B에 차량 감지  (T=Y for 1 sec) |
|  | Road B의 차량 이동  (T=R for 1 sec) |
|  | Road B의 차량 이동  (T=R for 1 sec) |
|  | Road A 신호 초록불  (T=G for 1 sec) |

State는 위와 같이 , 총 5개로 구성하였다. Road B에 차가 없을 때는 Road A의 신호등은 초록불을 유지한다. 이 후 Road B에 차가 감지되면 신호등은 노란불을 1초 동안 유지한 후 빨간불을 2초 동안 유지한다. 이 때 2초간 점등하는 빨간불과 초록불이 1초간 점등하는 시간으로 State로 각각 두 개의 State로 나누었다. 다음으로는 Road A의 신호등이 초록불을 2초간 점등한 후 Road B에 차량이 있는지 유무에 따라 State가 진행된다.

### State Graph / State Table

도표, 원, 텍스트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

Figure 1. Moore State Graph of Traffic Light Controller

|  |  |  |  |
| --- | --- | --- | --- |
| **Present State** | **Next State** | | **Output** |
|  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

앞서 정의한 State를 통해 우리는 위와 같이Moore State Graph를 그려 Input에 의한 State 변화를 확인하였고 이를 State Table로 제작하였다.

### Logics for Next-State and Outputs

|  |  |  |  |
| --- | --- | --- | --- |
| **Present State**  (ABC=) | **Next State** (ABC=) | | **Output** |
|  |  |  |
| 000 | 000 | 001 |  |
| 001 | 010 | 010 |  |
| 010 | 011 | 011 |  |
| 011 | 100 | 100 |  |
| 100 | 000 | 000 |  |

위의 Table은 앞서 제작한 State Table을 FSM으로 제작하기 위해 Binary로 변환하였으며 3 bit의 Present State와 1 bit의 input (V)를 통해 아래와 같이 Next State의 와 출력(G,Y,R)의 Karnaugh Map을 제작하였으며 minimum SOP를 구하였다. 이 때 우리는 까지 State를 정의하였으므로 ABC의 101 이상의 수는 Don’t Care Value로 설정하였다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **CV**  **AB** | **00** | **01** | **11** | **10** |
| **00** | 0 | 0 | 0 | 0 |
| **01** | 0 | 0 | 1 | 1 |
| **11** | X | X | X | X |
| **10** | 0 | 0 | X | X |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **CV**  **AB** | **00** | **01** | **11** | **10** |
| **00** | 0 | 0 | 1 | 1 |
| **01** | 1 | 1 | 0 | 0 |
| **11** | X | X | X | X |
| **10** | 0 | 0 | X | X |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **CV**  **AB** | **00** | **01** | **11** | **10** |
| **00** | 0 | 1 | 0 | 0 |
| **01** | 1 | 1 | 0 | 0 |
| **11** | X | X | X | X |
| **10** | 0 | 0 | X | X |

|  |  |  |
| --- | --- | --- |
| **C**  **AB** | **0** | **1** |
| **00** | 1 | 0 |
| **01** | 0 | 0 |
| **11** | X | X |
| **10** | 1 | X |

|  |  |  |
| --- | --- | --- |
| **C**  **AB** | **0** | **1** |
| **00** | 0 | 1 |
| **01** | 0 | 0 |
| **11** | X | X |
| **10** | 0 | X |

|  |  |  |
| --- | --- | --- |
| **C**  **AB** | **0** | **1** |
| **00** | 0 | 0 |
| **01** | 1 | 1 |
| **11** | X | X |
| **10** | 0 | X |

### Simulation Circuit Design

도표, 평면도, 기술 도면, 개략도이(가) 표시된 사진

자동 생성된 설명

Figure 2. Traffic Light Controller Whole Circuit (Logisim)

도표, 기술 도면, 평면도, 개략도이(가) 표시된 사진

자동 생성된 설명

Figure 3. Next\_State Circuit (Logisim)

도표, 라인, 스케치, 기술 도면이(가) 표시된 사진

자동 생성된 설명

Figure 4. Output Circuit (Logisim)

폰트, 도표, 스크린샷, 텍스트이(가) 표시된 사진

자동 생성된 설명

Figure 5. Simple Circuits (Logisim)

텍스트, 도표, 평면도, 기술 도면이(가) 표시된 사진

자동 생성된 설명

Figure 6. Block Circuits (Logisim)

도표, 기술 도면, 스케치, 평면도이(가) 표시된 사진

자동 생성된 설명

Figure 7. Updating Next State Circuit (Logisim)

앞서 구한 minimum SOP를 가지고 Logisim 프로그램을 활용하여 회로를 제작하였다. 전체적인 회로의 모습은 Figure 2에서 볼 수 있듯이 5개의 AND gate, 1개의 XOR gate, 1개의 OR gate로 이루어짐을 확인할 수 있다. 이 후 FSM Block Circuit을 제작하기 위해 Next\_State Circuit과 Ouput Circuit으로 나누어 제작하였다 (이는 Figure 3과 Figure 4에서 확인할 수 있다.) 이를 가지고 Figure 6의 Block Circuit을 제작하였으며 Input으로는 V와 CLK 신호가 들어오고 Next State Circuit을 통해 Update가 된 A, B, C가 Update 파트의 D-FF을 통해 updating을 진행한다. 그리고 Output Circuit으로 들어간 B와 C는 초록불, 노란불 혹은 빨간불(T=G,Y,R)을 출력한다. Figure 7은 Figure 6에 사용된 Next State Circuit의 모습이다.

# **Results and Discussion**

### Results

아래의 표는 Logisim Simulation을 이용한 상황에 따른 Input에 의한 출력 변화를 나타낸 것이다.

|  |  |
| --- | --- |
|  | Road B에 차가 없는 상황  (V=0)  신호등 : 초록불 유지  (T = 1 0 0) |
|  | Road B에 차가 감지된 상황  (V=1)  신호등 : 노란불 점등  (T = 0 1 0) |
|  | Road B에 차가 이동하는 상황  (V=1)  신호등 : 0~1초 동안 빨간불 점등  (T = 0 0 1) |
|  | Road B에 차가 이동하는 상황  (V=1)  신호등 : 1~2초 동안 빨간불 점등  (T = 0 0 1) |
|  | Road B의 차가 이동하지 않는 상황  (V=1)  신호등 : 1초간 초록불 유지  (T = 1 0 0) |
|  | Road B의 차량이 이동을 완료한 후 더 이상 감지되지 않을 때  (V=0)  신호등: 0~1초 동안 빨간불 점등  (T = 0 0 1) |
|  | Road B의 차량이 이동을 완료한 후 더 이상 감지되지 않을 때  (V=0)  신호등: 1~2초 동안 빨간불 점등  (T = 0 0 1) |
|  | Road B에 차량이 감지되지 않음  (V=0)  신호등 : V=1 전까지 초록불 유지  (T = 1 0 0) |

### Discussion

해당 보고서는 Traffic Light Controller를 Moore FSM으로 설계하였다. State의 수를 줄이는 것이 Logic Circuit의 복잡성을 줄이고 회로 제작 단가를 낮출 수 있으므로 State를 최소의 개수인 5개로 설정하였다. 이를 위해 입력의 변화에 따른 State 변화와 Output을 관찰하기 용이한 Moore로 선택하여 설계하였다. 본 Controller의 조건은 2초간 빨간불과 초록불을 유지해야하는데 주어진 Input은 CLK과 Road B의 차량 유무이므로 2초의 빨간불과 초록불의 State를 각각 1초씩으로 나누어 State를 정의하는 것이 핵심이었다. 또한 Update와 Combinational Circuit의 관계를 한눈에 보기 용이하게 하기 위해 Next State 회로와 Output 회로의 Combinational Circuit Block과 D-FF을 이용한 Updating Block을 구분하여 하나의 Block Circuit으로 제작하였다. 이 후 제작된 회로를 통해 Road B의 차량의 유무에 따른 출력을 확인하여 설계한 Traffic Light Controller FSM이 정상적으로 작동하는지 확인하였다.

FSM은 서술적으로 표현할 수 있는 모든 상황을 입출력과 State 정의를 통해 하나의 Logic으로 만들 수 있다는 점에서 매우 큰 장점을 가지고 있다. 따라서 Input과 Output을 정확하게 규정하고 State를 효율적이며 정확하게 정의한다면 나만이 가질 수 있는 Logic을 설계할 수 있음을 깨달았다.

### Demo Video

[Video Link](https://www.youtube.com/watch?v=HqeGxrcn1_g&ab_channel=%ED%95%9C%EB%8F%99%EB%8C%80%ED%95%99%EA%B5%90%EC%95%88%EA%B2%AC%ED%9E%90)

# **Appendix**