1. Abstract

이번 프로젝트에서는 한학기 동안 배운 Simplification 방법과, TTL Chip의 사용법, Bread Board를 이용한 Debugging 및 Design 방법 등을 최대한 이용해서 Combinational Circuit과 Sequential Circuit을 작성하여 최종적으로 5 state 이상의 Finite State Machine을 구현할 것이다.

이 프로젝트는 동물의 종류를 정하여 알을 부화시키고 몇 마리 인지 기록하는 시스템으로 일종의 동물농장을 만들어 보는 것이다. 알이 받는 관심의 정도(앞으로 이 프로젝트에서는 '애정도' 라고 부르기로 한다.)를 3가지의 다른 방법으로 증가시킬 수 있는데, 각각 애정도의 증가량이 다르다. 정해진 애정도를 충족시키면 알을 부화시킬 수 있고, 만약에 먼저 부화시키고 싶다면 적당량의 애정도를 이용해 조기 부화 시킬 수 있다. 총 7 state로 구성되어 있고, Input은 애정도 증가에 3개, 조기 부화에 1개, 그리고 초기화에 1개가 필요하다. 이 과정을 Finite State Machine으로 구현할 것이며 counter와 adder 등을 주로 이용하여 구현한다.

2. Theoretic Backgrounds

2.1. Multi-level Simplification

Simplification의 방법 중 하나로 Quine-McCluskey는 논리식을 최소화하는 방법이다. 원리는 K-Map과 동일하지만, 약간 방법은 다르다. 컴퓨터에서 Simplification을 하기 위해서 표를 사용하는 Quine-McCluskey 방법을 만들었고, 이를 이용해 Simplification을 할 수 있다.

K-Map과 Quine-McCluskey 방법을 통해서 Simplification을 하게 된다면 대부분 최적의 회로로 만들 수 있다. 그런데 회로가 복잡할 경우에는 Two-level로 Simplification만으로는 최적의 결과를 만들어 내기 어려워 또 다시 Boolean Algebra를 이용해 추가적인 Simplification을 해야한다. 이럴 경우에는 특별한 방법이 존재하는 것이 아니라, 주로 Boolean Algebra의 Axiom들을 이용하여 최소한의 Term들만 남도록 식을 변형하는 것이다.

2.2. Finite State Machine (FSM)

State라는 것은 Sequential Circuit에서 기억소자들이 가지는 출력 값을 나타낸 것으로 하나의 상태를 지칭하는 것이다. 이러한 State들이 특정 개수로 유한하기 때문에 Sequential Circuit을 Finite State Machine이라고 부르기도 한다. Finite State Machine에서 State는 Clock이 바뀔 때 마다 Transition이 발생하는데, 이를 한눈에 보기 위해서 표나 그림으로 표현할 수가 있다. 이를 디자인하기 위해서는 State Transition Diagram과 State Transition Table을 그려내고 도출해내는 과정이 필요하다. 그래야 전체적인 구조의 흐름을 알 수 있고, Present state에서 Next state로 변할 때의 Input들을 이용하여 Simplification을 진행할 수 있다. 그렇게 된다면 Input Equations를 정할 수가 있고 이를 이용해서 최종적으로 Design할 수가 있게 되는 것이다.

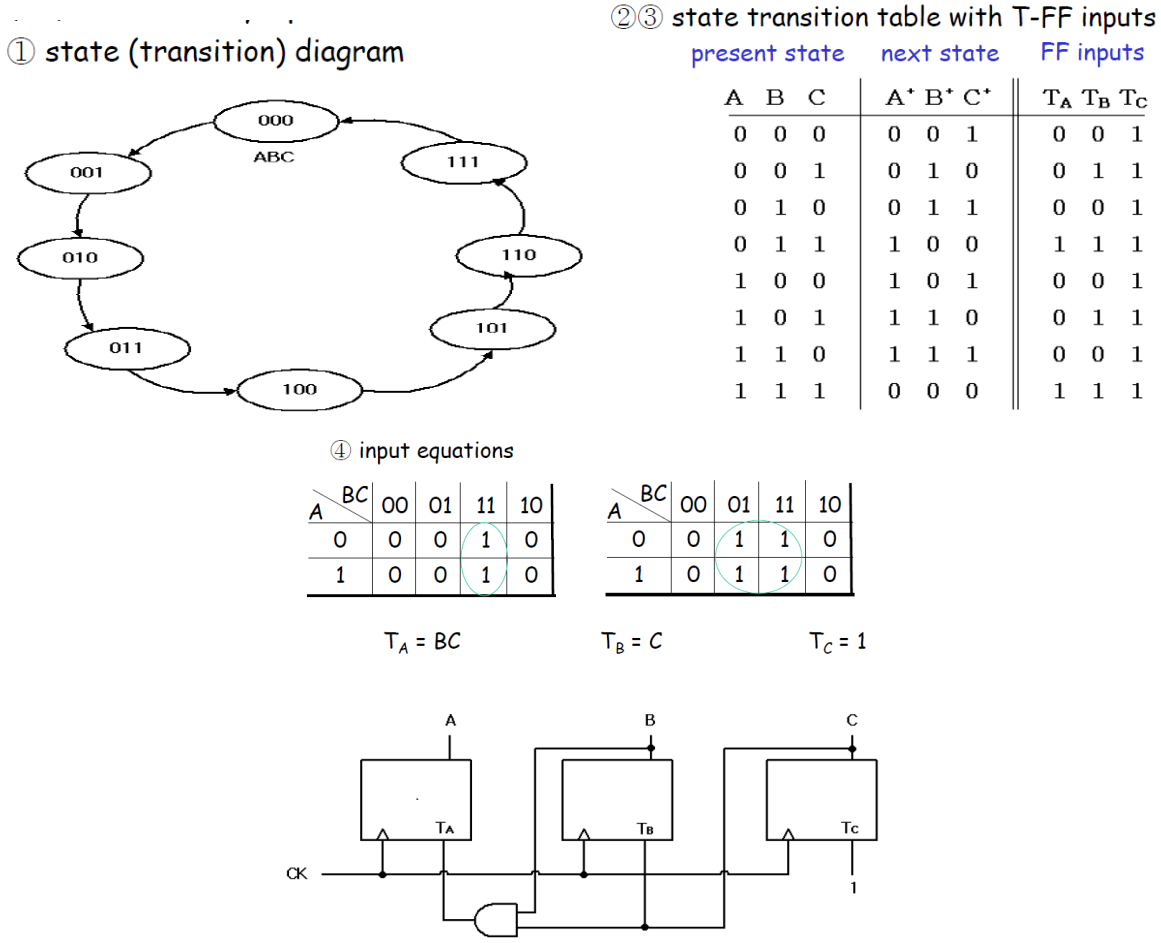


Figure. FSM Design 과정 예시(Counter)

2.3. 7-Segment LED & BCD-to-7 Segment Decoder

7-Segment LED란 각종 숫자 Display에서 널리 활용이 되고 있는 출력 부품으로서, a부터 g까지 7개의 막대에 불을 키고 끄고 하여 숫자를 나타내고 한 개의 Dot 입력으로 소수점을 나타낼 수 있다. LED는 크게는 Common Anode와 Common Cathode의 2가지 방식으로 나뉜다. Common Cathode는 2개의 전원에 모두 Ground를 연결하고 각각의 input에 3V의 전압을 인가하게 되면 LED에 불이 들어오게 된다. Common Anode는 2개의 전원에 모두 3V의 전압을 인가하고 a부터 g까지와 dot input에 Ground를 연결해야 LED에 불이 들어오게 된다.

실험실에서 사용하는 7-Segment LED를 포함하는 모든 LED의 정격 전압/정격 전류는 약 2V/20mA이다. LED는 Diode의 일종인데, 정격전압보다 약간 높은 전압이 인가된다면 과전류가 일어나 쉽게 고장이 날 수 있다. 그래서 저항 등을 사용함으로써 전압을 적정 수준으로 낮춰야 한다. 인가 전압은 5V이고 저항을 거쳐서 전압 하강이 3V 발생해야 하기에 옴의 법칙 (R = V/t)를 이용하여 계산하게 된다면 150Ω의 저항이 필요하게 됨을 알 수 있다. 150Ω 보다 크면서 최대한 가까운 저항을 사용하는 것이 바람직하다. 고휘도 LED는 정격 전압이 3V 정도이므로 80 ~ 100Ω

의 저항을 사용해야 하고 실험실에서는 안전 및 부품 수명의 문제로 약 200 ~ 400Ω 정도의 저항을 사용한다.

BCD-to-Seven Segment Decoder란 BCD Code로 주어진 4개의 input에 대하여 Seven Segment LED의 각 Segment에 해당되는 7개의 Input으로 변환해 주는 회로이다. 즉 이 회로는 A, B, C, D 4개의 input을 받아서 a, b, c, d, e, f, g 7개의 Output을 출력해주는 회로이다.

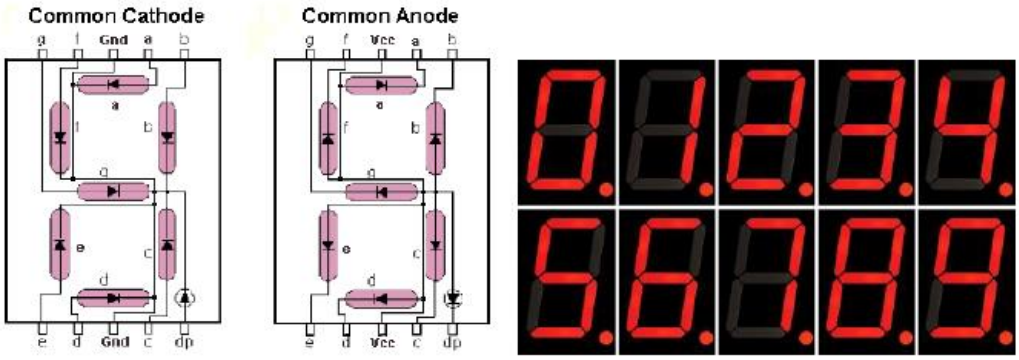
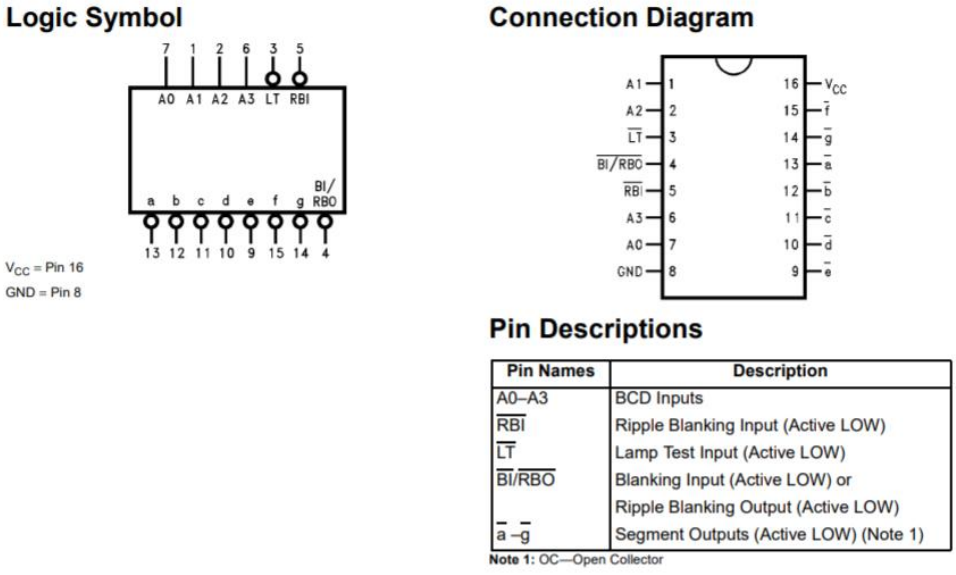


Figure 1. 7-Segment Decoder LED

다음은 BCD-to-Seven Segment Decoder를 구현한 74LS47 Chip의 Datasheet의 일부이다.



Pin Descriptions

Pin Names	Description
A0–A3	BCD Inputs
$\overline{RBI}$	Ripple Blanking Input (Active LOW)
$\overline{LT}$	Lamp Test Input (Active LOW)
$\overline{BI/RBO}$	Blanking Input (Active LOW) or Ripple Blanking Output (Active LOW)
$\overline{a}$ –$\overline{g}$	Segment Outputs (Active LOW) (Note 1)

Note 1: OC—Open Collector

Figure 2. 74LS47 Datasheet

이번 프로젝트에서는 종류가 무엇인지, 현재 애정도는 얼마나 되는지, 부화한 알의 개수는 얼마

인지, 그리고 조기 부화한 알의 개수는 얼마인지를 LED로 표시할 것이다.

2.4. T-Flip Flop

SR-Latch는 S와 R의 Input이 1에서 0으로 변하게 되면 Race Condition이 일어난다. S와 R의 Input이 1인 Forbidden Input을 없애기 위해서 추가로 AND Gate를 사용하여 J와 K를 인가하는 JK-Flip Flop을 구현할 수 있다. 여기서는 J와 K에 1이 인가 될 경우 Q가 0에서 1 혹은 1에서 0으로 바뀌는 Toggle 현상이 일어나게 된다. T Flip-Flop같은 경우는 J와 K의 값을 동시에 넣는 것으로 보통 Q의 값이 바뀔 때 T Input을 1로 한다.

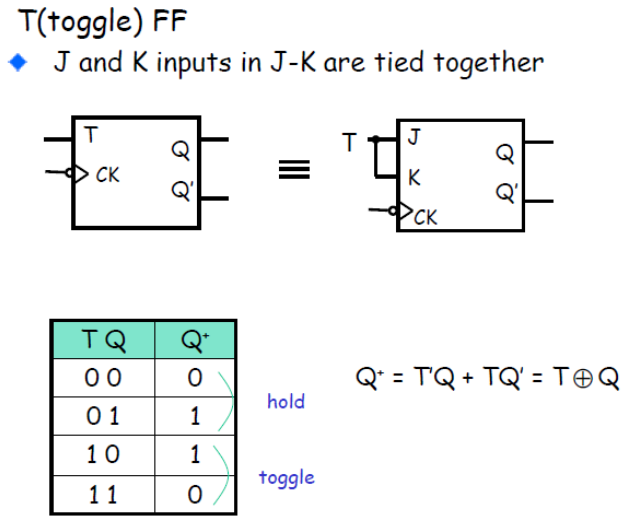


Figure 3. T-Flip Flop의 회로 및 특성표

2.5. Counter

Counter란 Clock에 따라 미리 정해진 숫자들을 저장 및 출력하는 회로로, 특수한 레지스터의 일종이다. Counter는 크게 Ripple Counter와 Synchronous Counter로 나뉜다. Ripple Counter인 경우 첫번째 Flip-Flop만 외부 Clock이 사용되는 Asynchronous Counter이다. 모든 후속 Flip-Flop들은 앞선 Flip-Flop의 출력에 의해 Clocking 된다. Ripple Counter의 Mod는 n개의 Flip-Flop이 사용되는 경우 2^n인데, 4비트의 카운터의 경우 0000 ~ 1111이다.

이번 프로젝트에서는 애정도를 카운트 업, 카운트 다운 시키거나 부화한 알의 개수를 카운트 업 하는데 사용할 것이다.

3. Project Setup

3.1 Circuit

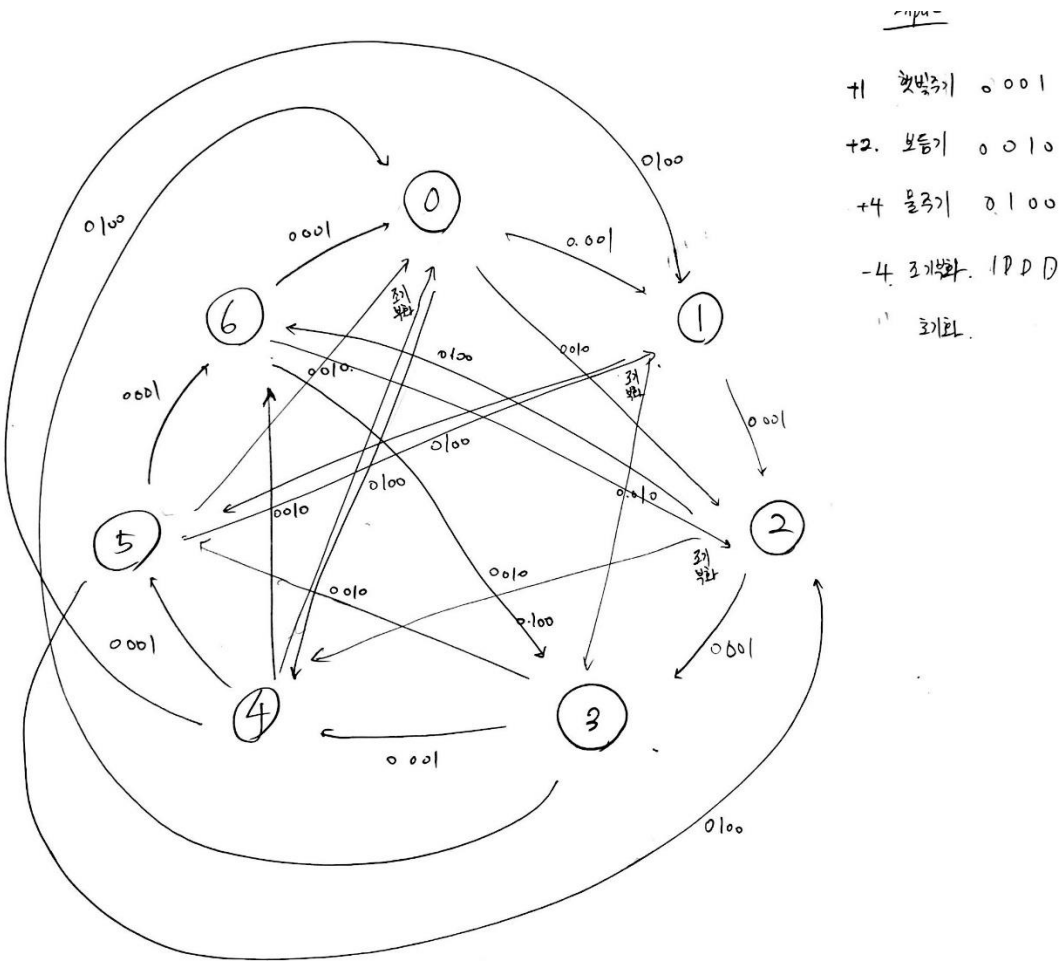
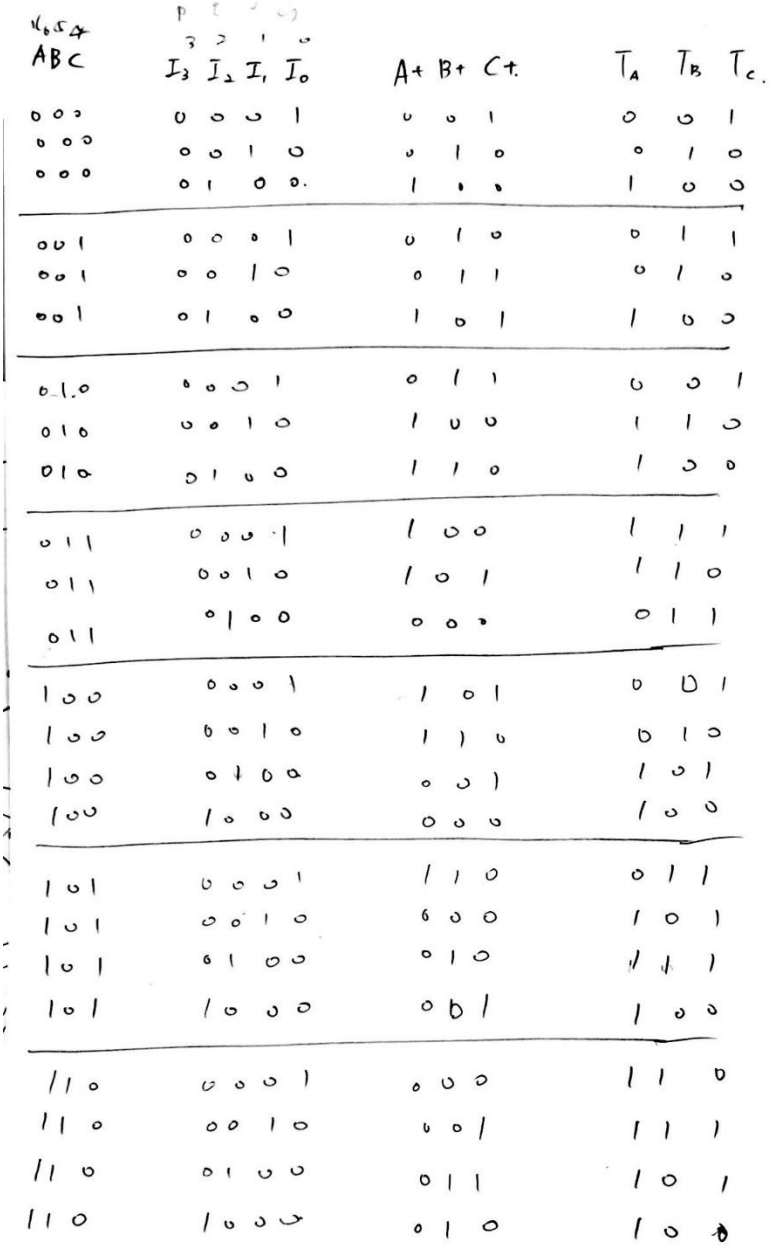


Figure 4. State Diagram

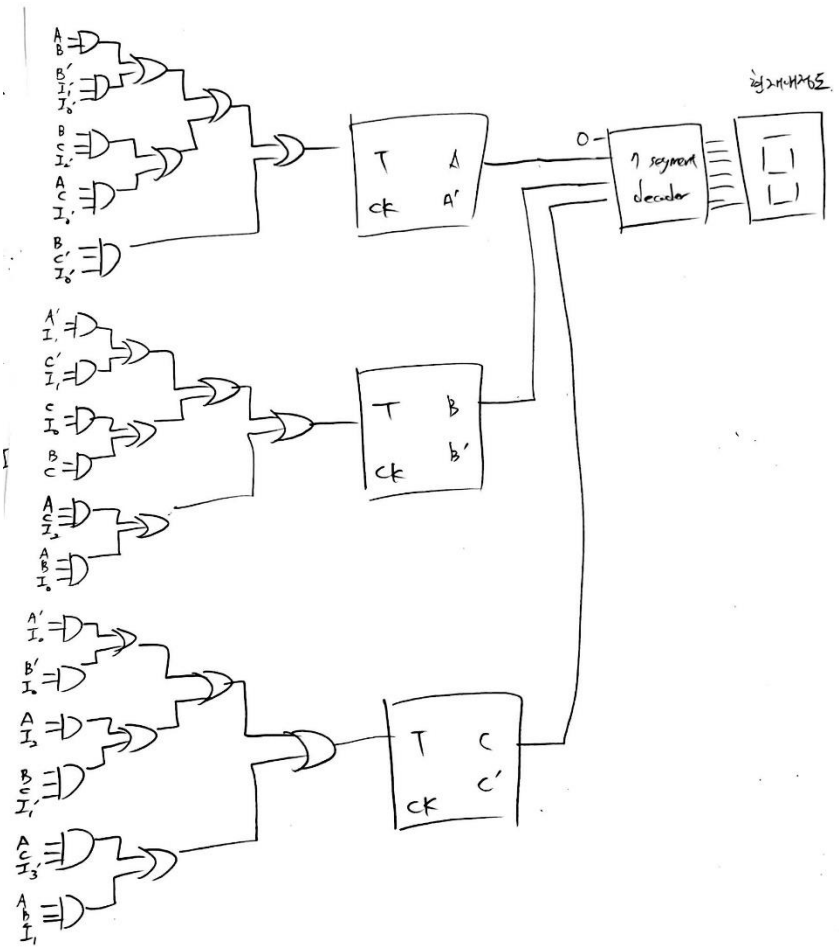


$$T_A = AB + B'I_1'I_0' + BCI_2' + ACI_0' + BC'I_0'$$

$$T_B = A'I_1 + C'I_1 + CI_0 + BC + ACI_2 + ABI_0$$

$$T_C = A'I_0 + B'I_0 + AI_2 + BCI_1' + ACI_3' + ABI_1$$

Figure 5. state transition table with T-FF



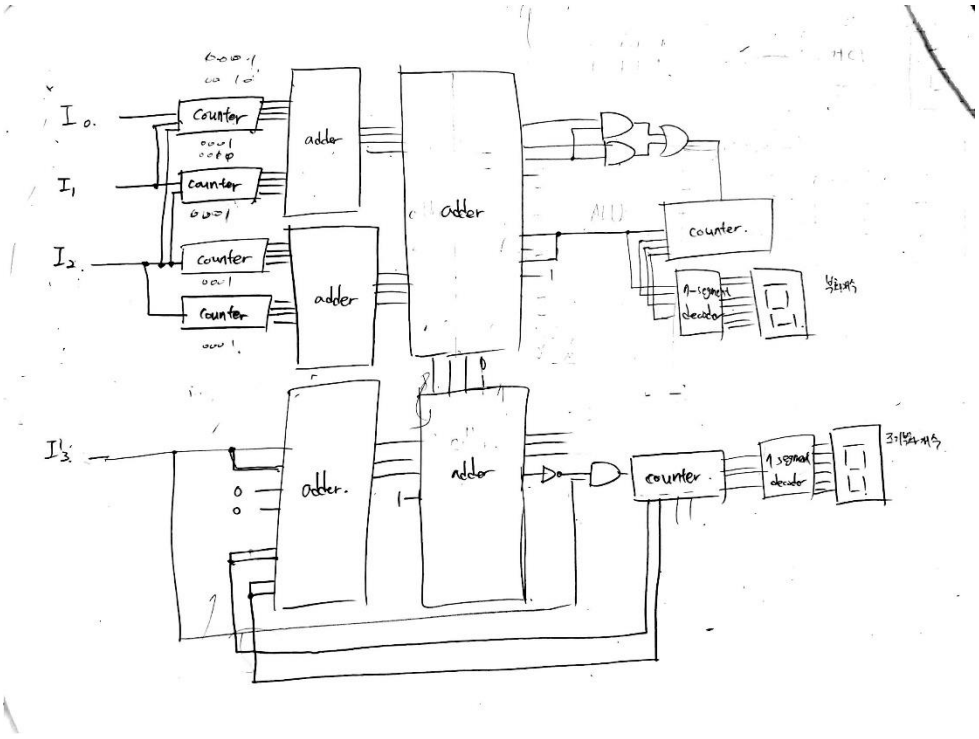


Figure 6. FSM Circuit

이번 프로젝트에서 구현할 FSM은 옛날에 아이들이 가지고 놀던 '다마고치'랑 유사하게 알을 키워 부화시키는 일종의 동물농장이다. 동물의 애정도는 0부터 시작하여 1, 2 혹은 4가 증가하는데 만약에 7에 도달하게 된다면 자동으로 부화하는 시스템이다. 만약 조기 부화를 시키고 싶다면 애정도 4을 소모해서 조기 부화 시킬 수 있으며, 애정도는 햇빛을 주거나 물을 주거나 보듬기를 하게 되면 상승하게 된다. Input으로 들어가는 '물주기', '햇빛'과 '보듬기'는 BCD Counter들을 통해 카운트가 된다. 이를 Adder를 연결하여 자릿수 조절을 잘 하여 1, 2, 혹은 4가 증가하도록 더한다. 7이 되는지는 Checker를 통해서 확인할 것이고 자동으로 부화 개수가 1이 증가할 것이다. 애정도를 7로 나눈 몫을 체크하여 Counter로 계수하여 개수를 늘리고, 중간에 애정도 4이상이 쌓였을 경우 4를 차감하여 조기 부화 개수를 증가시킬 것이다. 중간에 Subtractor는 조기 부화 Input을 체크하여 현재 애정도에서 차감했을 시 음수가 아니라면 조기 부화 개수를 증가시키는 것이다.

3.2 Inputs

이번 프로젝트에서 구현할 회로에서 총 6개의 Input을 입력 받으며 각각은 보듬기, 햇빛, 조기부화, 초기화, 그리고 종류를 선택할 수 있는 버튼 2개이다.

- 물주기(0100) : 키우는 알의 애정도가 +4이 된다.
- 보듬기(0010) : 키우는 알의 애정도가 +2가 된다.
- 햇빛(0001) : 키우는 알의 애정도가 +1이 된다.
- 조기부화(1000) : 키우는 알의 애정도를 -4이 되며 이는 애정도가 4, 5 혹은 6일 때 가능하다.
- 초기화 : 초기상태로 초기화 한다.

State부터 Input까지 우선 7가지의 Input을 0과 1을 부여함으로써 각각의 상황을 주어지게 한다. 처음에는 앞의 3자리는 0부터 시작하므로 000에 놓은 뒤에 뒤의 4자리 가지고 애정도를 증가시키거나 감소시키면 된다.

3.3 Outputs

이번 프로젝트에서 구현할 회로에서는 총 3가지 정보를 Output으로 출력할 수 있으며, 현재 애정도, 조기 부화한 알의 개수, 부화한 알의 개수이다.

- 현재 애정도 : 현재 부화 전 알의 애정도가 0부터 6까지 표시가 된다. 여기서 주목할 점은 현재 애정도가 7이 된다면 자동으로 0으로 초기화가 되어 계속해서 숫자가 더해지는

식이다. 만약에 10이 된다면 7을 소모하여 LED에는 3만 표시되어 있을 것이다. 현재 애정도 같은 경우 결국에는 2가지의 기능이 내장되어 있다고 볼 수가 있다.

- 조기 부화한 알의 개수 : 현재 애정도를 4을 감소시키면 조기 부화한 알의 개수가 1이 증가하며, 몇 개가 조기 부화하였는지 카운트하여 볼 수가 있다.
- 부화한 알의 개수 : 현재 애정도가 7 혹은 7를 넘었을 경우에 부화한 알의 개수가 1이 증가되며, 몇 개가 부화하였는지 카운트하여 볼 수가 있다.

< 필요 부품 >

- 74LS90(BCD COUNTER) 6개
- SN74LS283(4-bit Adder) 5개
- G-1056K(7 segment anode LED) 3개
- 74LS47(BCD to 7 segment decoder) 3개
- 74LS08(2-input and gate) 3개
- 74LS32(2-input or gate) 4개
- 74LS04(inverter) 2개
- 74LS11(3-input and gate) 4개
- 74LS73(dual JK Flip-flop) 2개
- 200ohm 저항 6개

4. Result

"알 부화"라는 게임을 성공적으로 만들었다. 애정도는 1부터 7까지 존재하지만, 7이되면 자동으로 알이 하나 부화하고 애정도는 다시 0이 된다. 총 4가지의 Input을 부여할 수 있다. 4가지 상태를 00, 01, 10, 11로 정의하여 LED에 불빛으로 주어진 Input이 어떠한지 확인이 가능하다. 00은 햇빛주기, 01은 보듬기, 10은 물주기, 그리고 11은 조기 부화 버튼인다. 그리고 이를 확인할 수 있는 Output으로 7-segment LED가 3개가 존재한다. 조기 부화한 알의 개수, 부화한 알의 개수, 그리고 현재 애정도를 숫자로 볼 수가 있다.

00상태인 햇빛주기에서는 Input LED의 불빛이 하나도 들어오지 않는다. 이를 시전하게 된다면 알의 현재 애정도가 1이 증가하여 LED의 값이 1 증가된 값이 보인다. 조기 부화한 알의 개수와 부화한 알의 개수는 증가하지 않는다. 만약 6에서 이를 시전하게 된다면 현재 애정도는 7이 되지만 LED는 0으로 자동 초기화가 되면서 부화한 알의 개수가 1개 증가하는 것을 볼 수가 있다.

01상태인 보듬기에서는 Input LED 중 오른쪽 불빛만 들어오고 이를 시전하게 된다면 알의 현재 애정도가 2가 증가하여 LED의 값이 2 증가된 값이 보인다. 조기 부화한 알의 개수와 부화한 알의 개수는 증가하지 않는다. 만약 5에서 이를 시전하게 된다면 현재 애정도는 0이 되면서 부화한 알의 개수가 1개 증가하고, 6에서 시전하게 된다면 현재 애정도는 1이 되면서 부화한 알의 개수가 이때에도 1개 증가한다.

10상태인 물주기에서는 Input LED 중 왼쪽 불빛만 들어오고 이를 시전하게 된다면 알의 현재 애정도가 4가 증가하게 된다. 현재 애정도가 3, 4, 5, 6에서 시전하게 된다면 각각 0, 1, 2, 3으로 바뀌면서 현재 애정도가 변하게 된다. 그러면서 부화한 알의 개수가 1개 증가하게 된다.

11상태인 조기 부화에서는 Input LED가 두개다 불빛이 들어오는데 이는 현재 알의 애정도를 4만큼 감소시킨다. 0, 1, 2, 3에서는 사용해도 아무런 변화가 되지 않는다. 4부터 사용이 가능하며, 이때에는 조기 부화한 알의 개수가 1개 증가하게 된다.

다음은 실제 데모를 진행했을 때 상황이다. 하얀색 Input 버튼이고 위에는 빨간 LED로 어떠한 버튼이 켜져 있는지 확인할 수 있다. 그리고 가장 왼쪽이 조기 부화한 알의 개수이고, 가운데가 부화한 알의 개수, 그리고 가장 오른쪽이 현재 애정도이다. 다음은 임의로 Ouput의 상태가 1 1 0 일 때의 상황이다.

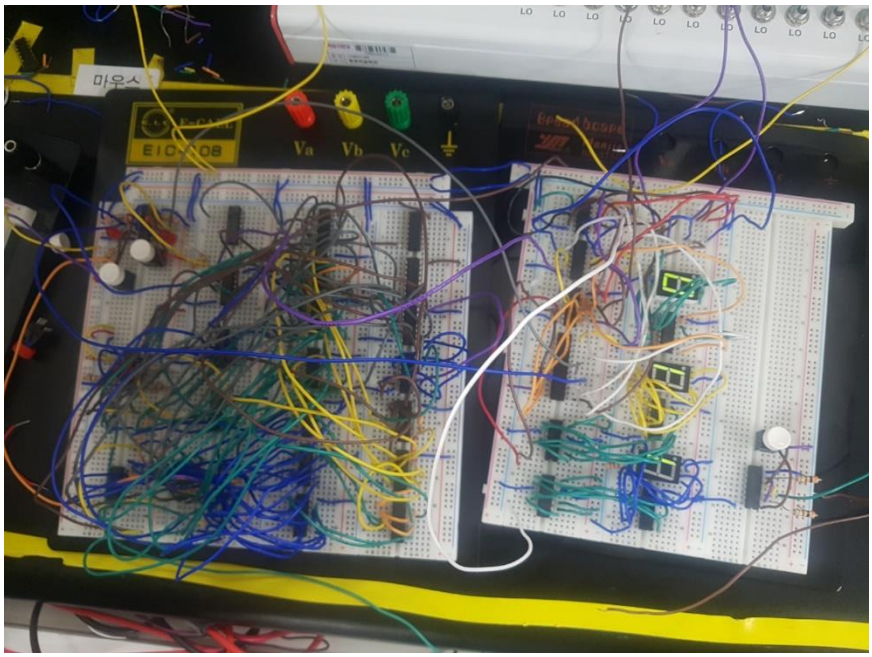


Figure 7. 데모 예시 1 1 0

Input 버튼을 둘 다 누르지 않으면, 햇빛 주기가 시전이 되어 현재 애정도가 1이 증가한다.

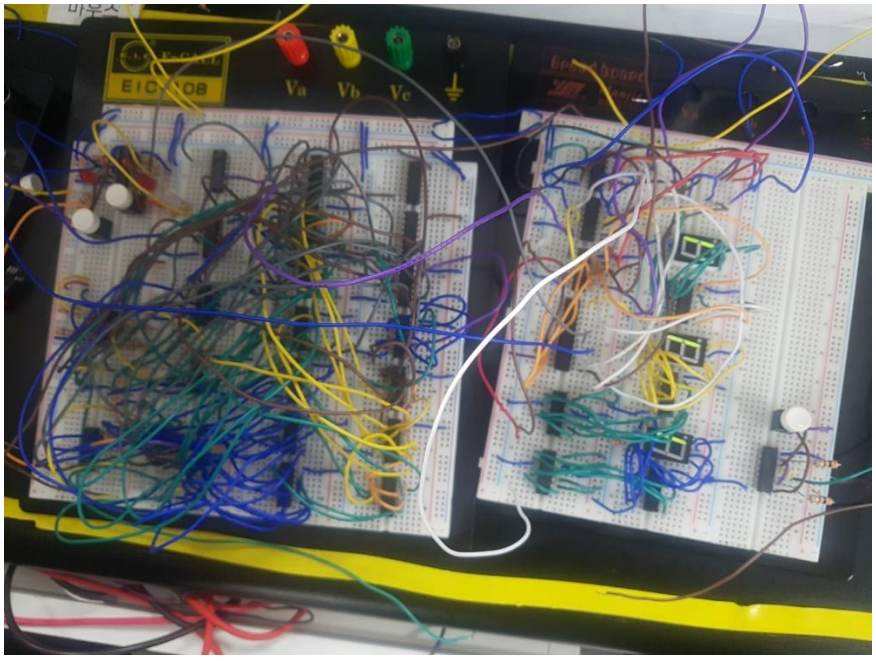


Figure 8. 데모 예시 1 1 1

오른쪽 버튼만 누르게 되면 보듬기가 시전되어 현재 애정도가 2가 증가한다.

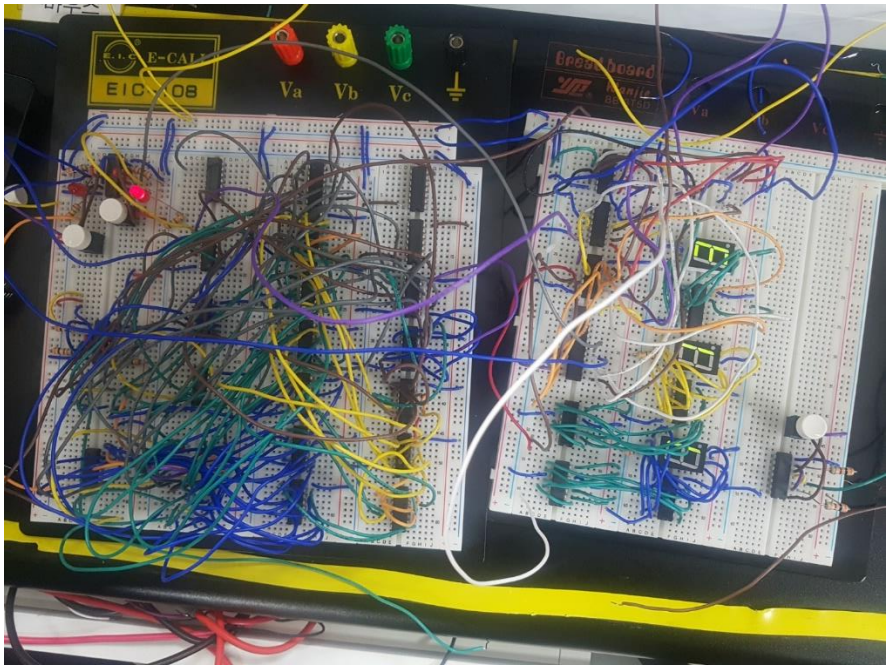


Figure 9. 데모 예시 1 1 3

왼쪽 버튼만 누르게 되면 물주기가 시전되어 현재 애정도가 4가 증가한다. 그리고 7이 넘게 되면 자동으로 부화한 알의 개수가 1이 증가함을 확인할 수가 있다.

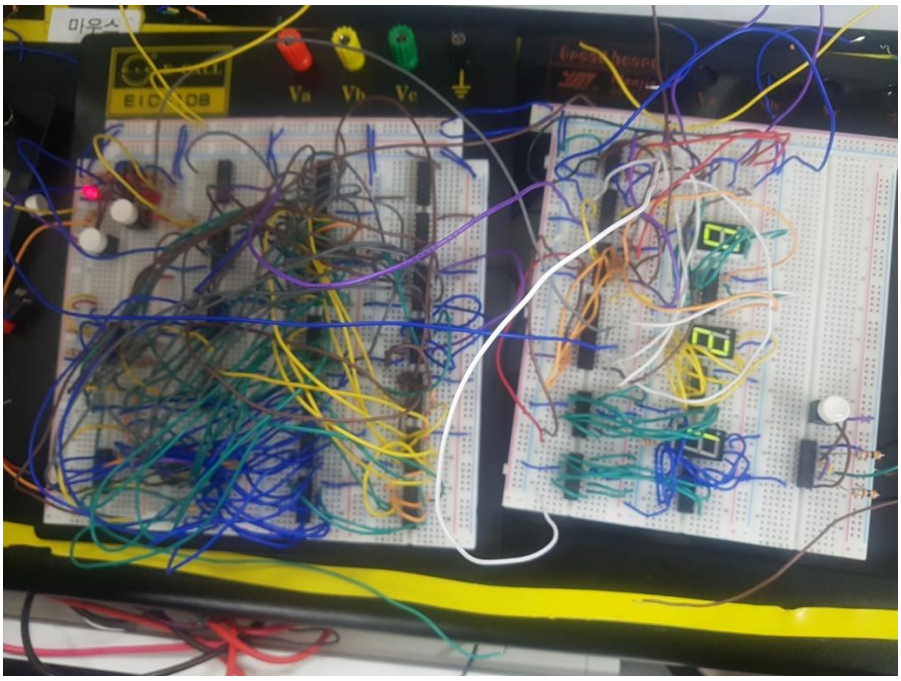


Figure 10. 데모 예시 1 2 0

물주기를 한번 더 시전해 현재 애정도가 4가 되도록 한다. 이때부터 조기 부화가 가능하다.

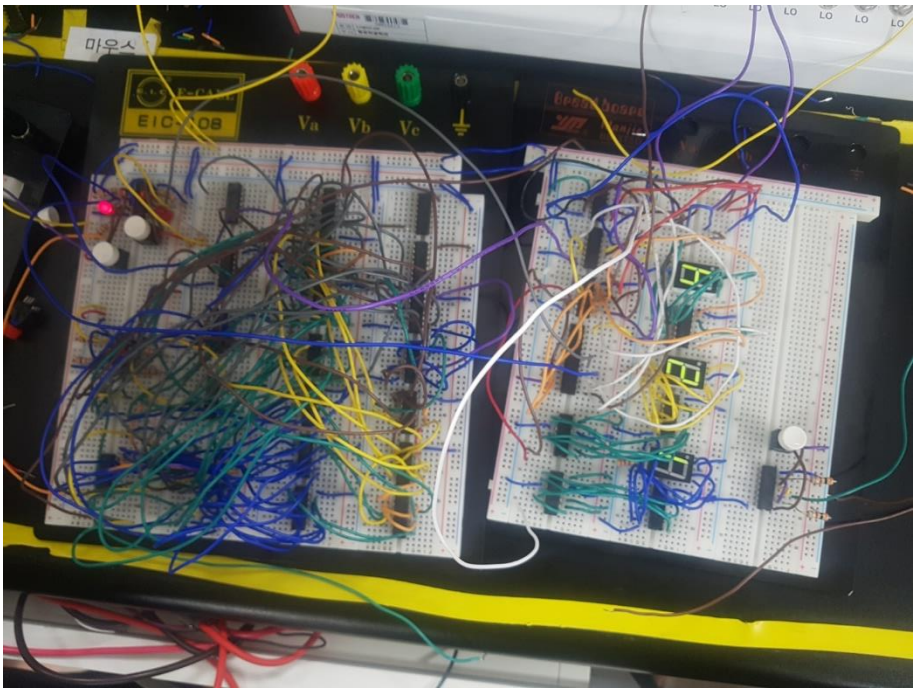


Figure 11. 데모 예시 1 2 4

흰색 버튼을 둘 다 누르고 현재 애정도가 4가 넘었을 때 조기 부화를 시전하면, 현재 애정도 4를 소모하여 조기 부화한 알의 개수를 1개 늘릴 수 있다.

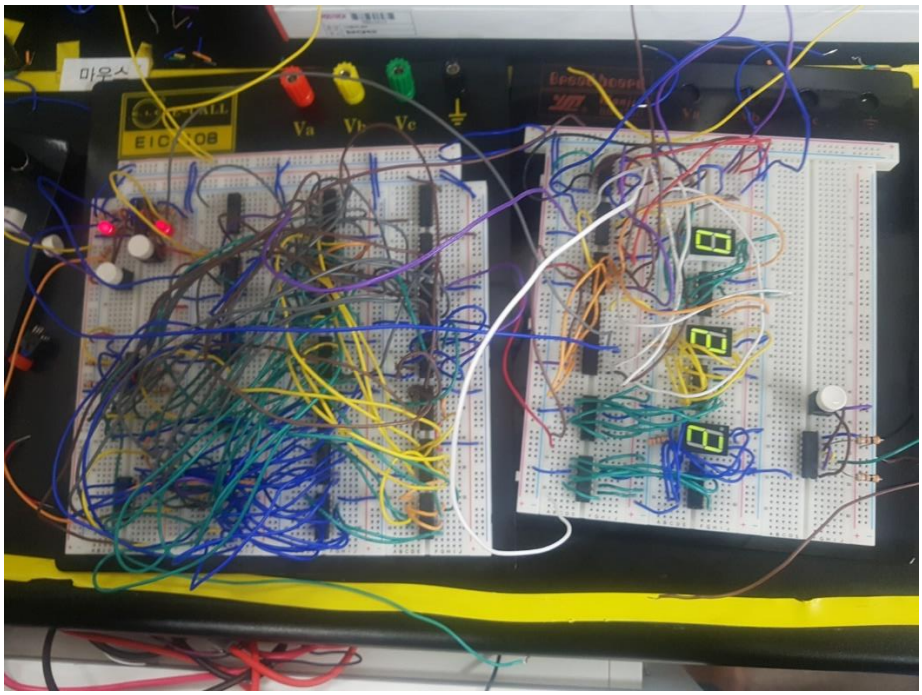


Figure 12. 데모 예시 2 2 0