**디지털회로실험 보고서**

-5주차-

전자공학과

2015104027

박정진

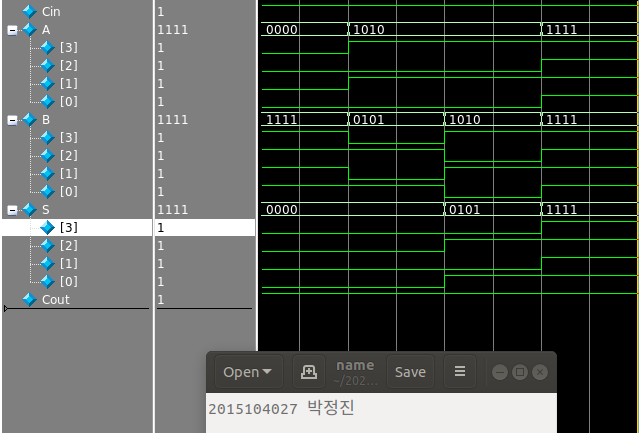
**실험 결과**

**Lab08 - 1 TTL74283**

A picture containing clock

Description automatically generated

A close up of a computer

Description automatically generated

A picture containing screenshot

Description automatically generatedA picture containing screenshot

Description automatically generated

첫 번째 결과 -> 3 + 2 + 0 = 5 (5'b00101)

두 번째 결과 -> 8 + 8 + 1 = 17 (5'b10001)

저번 당일결과 보고서에서 설명을 이상하게 하였는데 쿼터스 시뮬레이션에서는 틀린게 없다. 다만 Carry out을 결과의 MSB로 취급한다면 15+15+1 = 31(5' b11111) 로 즉 5bit의 결과는 모두 맞게 계산이 정확하다. 하지만 입력 비트수의 제한과 그에 따른 결과 비트수의 제한으로 계산 결과값은 0~31 까지 밖에 계산을 못한다는 것을 잘 못 표현했다.

**Lab08 - 2 ttl74283 Delay Time**

A picture containing screenshot, computer

Description automatically generatedA screenshot of a video game

Description automatically generatedA screenshot of a computer

Description automatically generated

VDD = 1.2V

|  |  |
| --- | --- |
| Model | Delay Time |
| fast, | 4.246ns |
| slow, | 7.424ns |
| slow, 85 | 8.308ns |

공급전원은 1.2V 고정으로 하고 Device 는 Cyclone 4, EP4CE115F29C7N를 이용했다. 그리고 해당 회로에 구성되어있는 MOSFET모델을 코너 별, 온도 별로 Delay 를 측정 해보았다.

예상대로 Fast 코너가 Slow 코너보다 Delay 가 적음을 알 수 있고 온도가 높을 수록 더 Delay Time 이 길어지는 것을 확인 할 수 있다. 이는 온도가 높아지면서 MOSFET의 electron, hole 들이Lattice scattering 에 영향을 더 많이 받아 mobility 가 감소한다. 이는 MOSFET 의 양이 적어지는 것을 의미한다. 전류량이 적어지면 MOSFET 의 속도가 줄어든다는 것을 뜻한다. (따라서 CMOS Logic gate에서는 공급전원 또한 전류량을 결정하는 요소이기 때문에 1.2V로 고정한 후 비교 분석했다. Fast Corner는 일반적으로 Power를 더 소모하므로 Power와 속도를 Trade-off로 상황에 따라 맞는 모델을 쓰면 됨을 알 수 있다.)

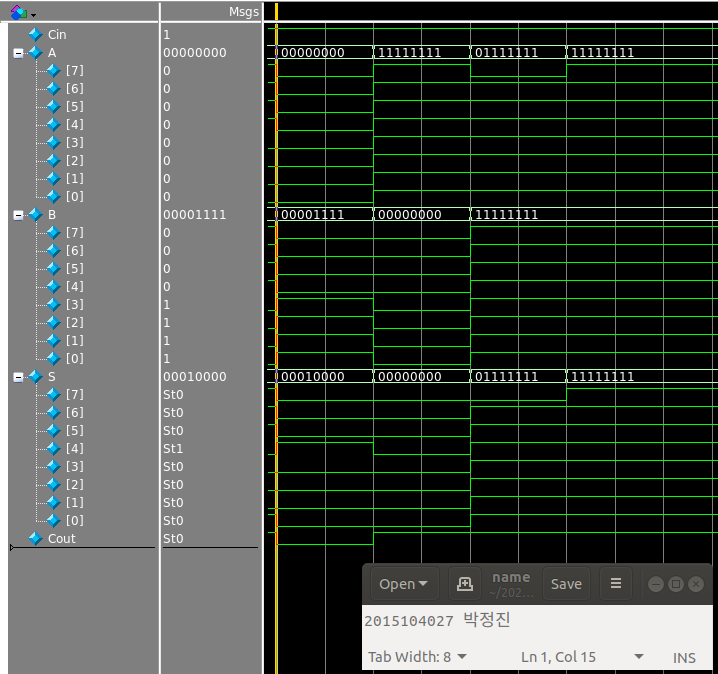
Delay 의 빵판 실험은 실험할 수 없다. 쿼터스의 gate-level simulation과 다르게 빵판 시뮬레이터는 딜레이에 대한 모델이 구현이 되어 있지 않기 때문에 딜레이 없이 ideal한 결과만을 볼 수 있기 때문이다.

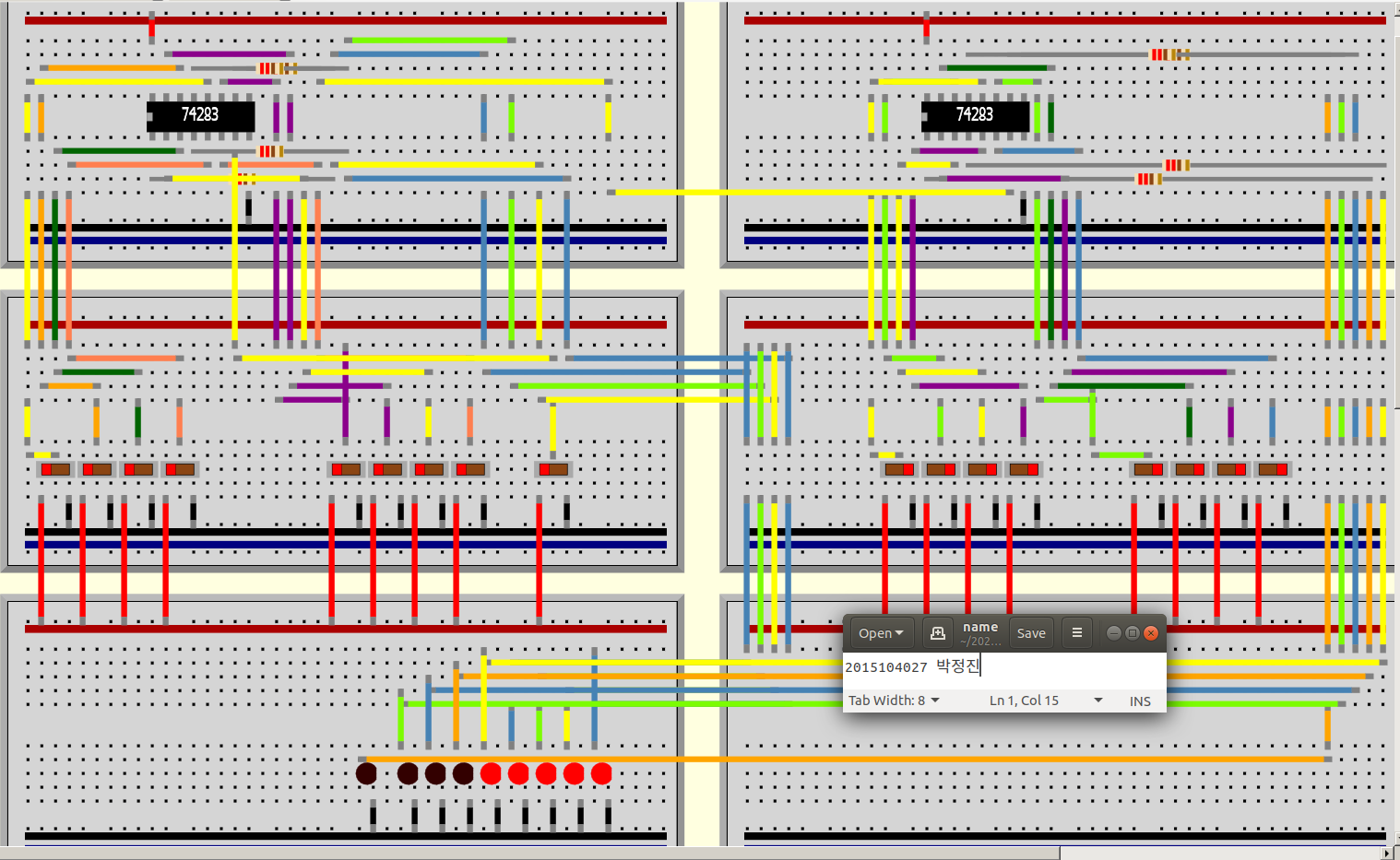
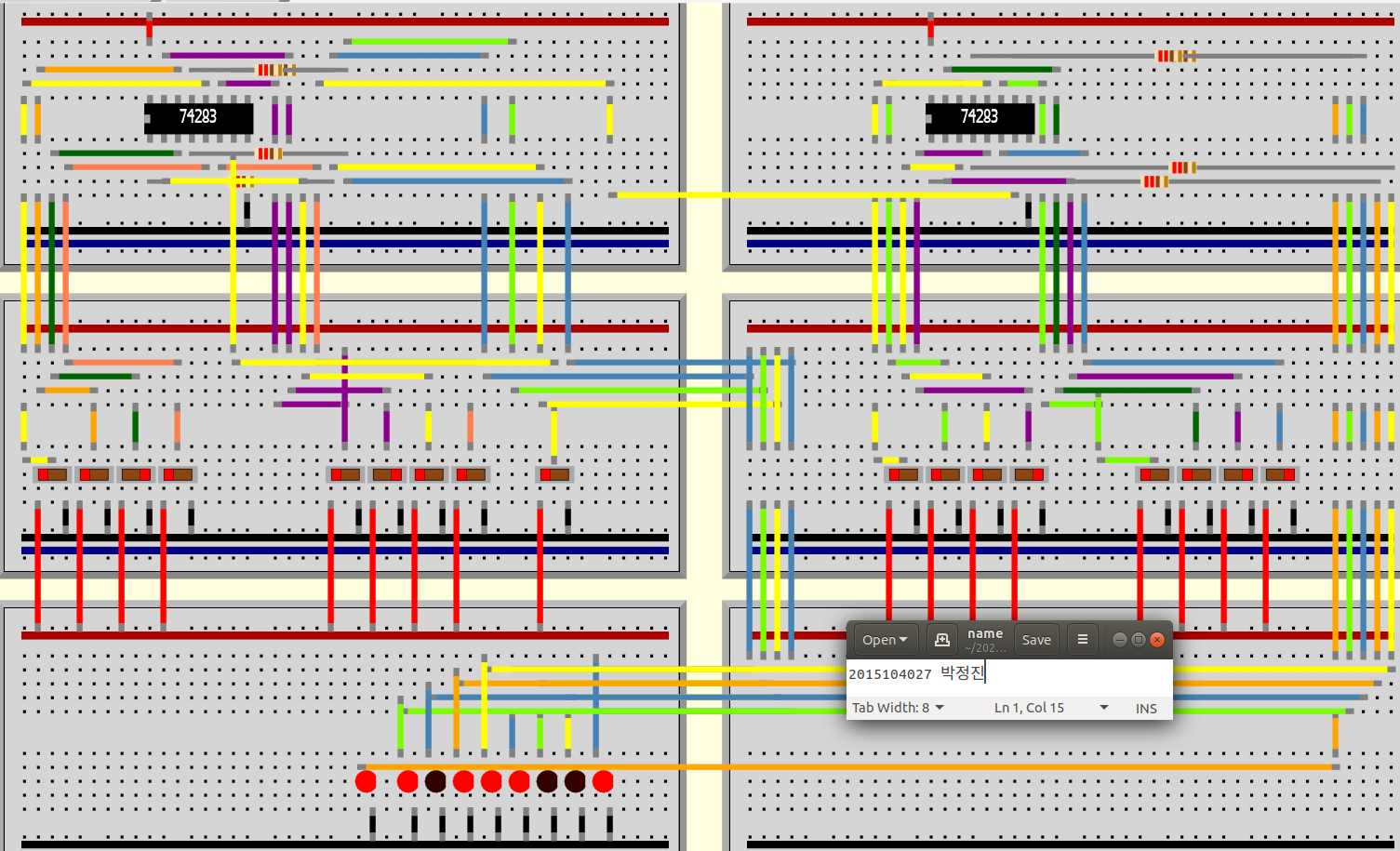
**Lab08 - 3 8bits adder**

A screenshot of a cell phone

Description automatically generatedA close up of a computer

Description automatically generatedA close up of a computer

Description automatically generated



첫 번째 그림 -> 237 + 203 + 1 = 441

두 번째 그림 -> 16 + 16 + 1 = 31

이 파트의 설명 또한 위와 동일하게 설명을 이상하게 하였다. 하지만 쿼터스 시뮬레이션 결과는 정확하고 여기서도 마찬가지로Carry out을 결과의 MSB로 취급한다면511로 즉 9bit의 결과는 모두 맞게 계산이 정확하다. 하지만 입력 비트수의 제한과 그에 따른 결과 비트수의 제한으로 계산 결과값은 0~511 까지 밖에 계산을 못한다는 것을 잘 못 표현했다.

**Lab08 - 4 8bits adder Delay Time**

A screenshot of a computer

Description automatically generated

A screenshot of a cell phone

Description automatically generatedA picture containing screenshot

Description automatically generated

VDD = 1.2V

|  |  |
| --- | --- |
| Model | Delay Time |
| fast, | 5.742ns |
| slow, | 10.596ns |
| slow, 85 | 11.736ns |

위의 실험과 마찬가지인 조건에서 실험을 진행했다. 결과는 예상한 그대로 fast가 slow 보다, 온도가 낮을 때가 높을 때보다 Delay time 이 적음을 확인 할 수 있다.

Delay time 이 이전 Adder를 한 개 썼을 때보다 Carry 를 Series로 연결 한 것이 Delay Time 이 더 오래 걸리는 것을 확인 할 수 있다. 하지만 두 개를 썼다고 Delay가 두 배가 아닌 이유는

첫 번째로 Delay는 Gate level에서 각 게이트의 Delay만을 영향을 받고 Carry는 그 전 Adder에서 계산 되기만 하면 그대로 뒤의 계산은 Carry가 거치는 게이트들의 Delay만 지나면 계산이 완료되고 두 번 째 Adder는 Carry를 내보낼 것이다. 즉 두 번째 Adder는 앞의 Adder가 계산을 완료하고 계산을 하는 것이 아니라 A+B계산은 둘다 동시에 한다는 의미이다. 단지 +Cin의 계산만의 Delay가 추가되는 것이다. (Carry propagation)

두 번째로logic effort, fanout effort와 parasitic delay를 고려한 linear RC Delay model에서 이 회로의 delay는 이전 ttl74283의 두 배가 아니라는 것을 알 수있다. 그에 대한 분석은 보고서의 양이 길어지므로 생략한다.

해당 딜레이를 줄이기 위해서 사용되는 대표적인 방법은 Carry-lookahead adder인데 딜레이는 줄이지만 complexity가 adder의 수에 비례하여 기하급수적으로 늘어난다는 단점이 있다.

위의 실험과 마찬가지로 Delay 의 빵판 실험은 실험할 수 없다. 쿼터스의 gate-level simulation과 다르게 빵판 시뮬레이터는 딜레이에 대한 모델이 구현이 되어 있지 않기 때문에 딜레이 없이 ideal한 결과만을 볼 수 있기 때문이다.

**Lab09 - 1 TTL74163**

A picture containing clock

Description automatically generatedA close up of a computer

Description automatically generated

각 핀에 대한 설명을 붙이면

CP = Rising Edge Trigger (Positive edge clock)

MRn(Clrn) = Synchronous clear (active Low(0))

CEP(ENP) = Enable Parallel

CET(ENT) = Enable Trickle

PEn(LDn) = Synchronous load (active Low(0))

TC(RCO) = Ripple Carry Out

이다.

Truth Table

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| MRn | CP | CEP | CET | Pen | Q | RCO | Operation |
| 0 | ↑ | x | x | x | 0 | 0 | Clear |
| 1 | ↑ | x | x | 0 | D | - | Load |
| 1 | ↑ | 1 | 1 | 1 | Q(t-1)+1 | 0 | Count |
| 1 | ↑ | 1 | 1 | 1 | 1111 | 1 | Carry out |
| 1 | - | 0 | 1 | 1 |  |  | Hold&Carry |
| 1 | - | x | 0 | 1 |  |  | Hold |

4bits Counter 가 위의 Truth table 처럼 제대로 작동되는 것을 확인 할 수 있고 모든 Operation 을 확인 할 수 있다.

A picture containing screenshot

Description automatically generatedA picture containing screenshot

Description automatically generatedA picture containing screenshot

Description automatically generated

로드 기능 포함 카운터의 동작이제대로 출력 되는 것을 LED를 통해 확인 할 수 있다.

**Lab09 - 2 8bits counter**

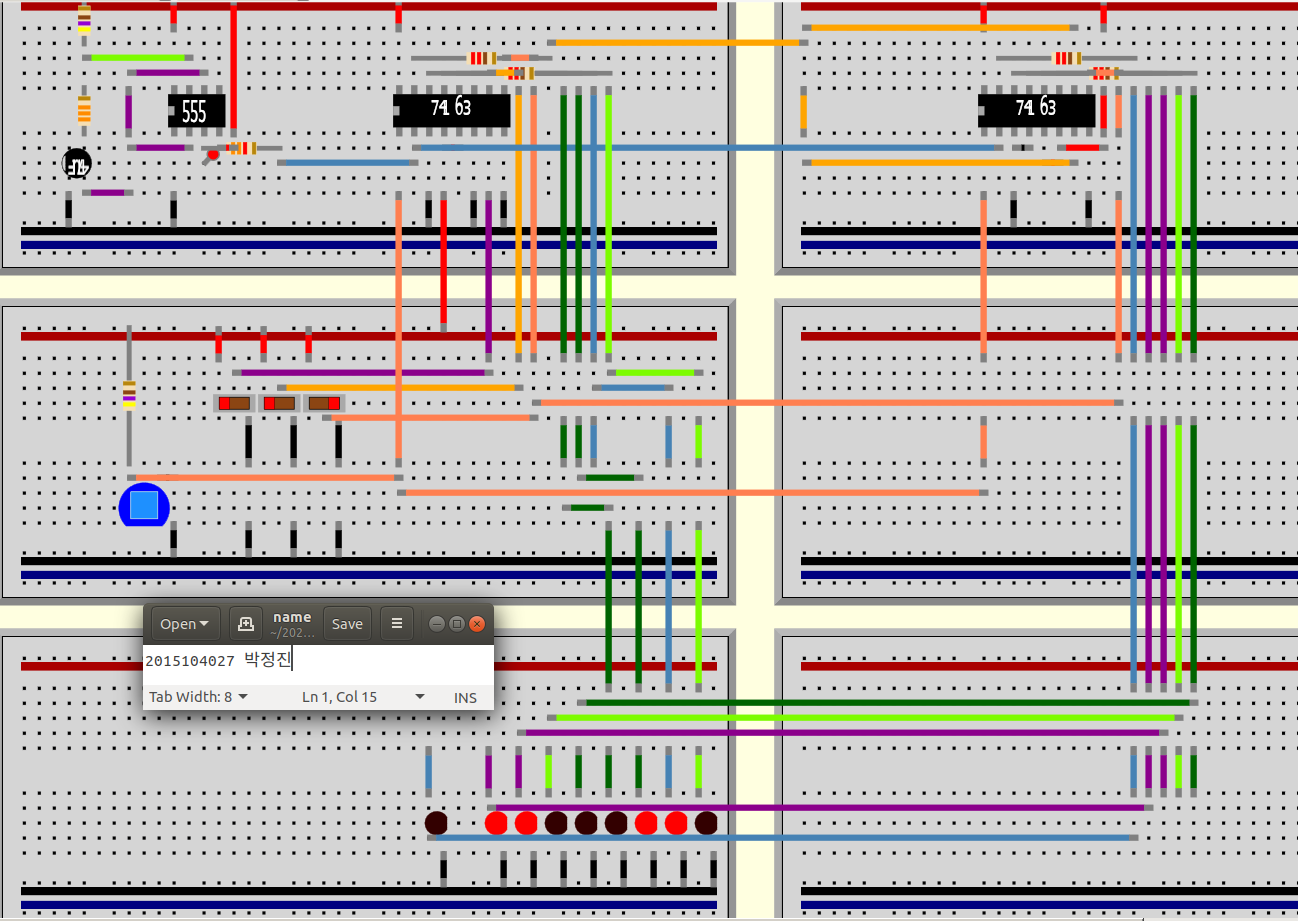
A screenshot of a cell phone

Description automatically generatedA close up of a computer

Description automatically generated

위의 실험과 마찬가지로 각 Truth table의 경우마다 동작이 완벽히 이루어짐을 확인할 수 있다. 두 ttl 의 연결은 LSB쪽 [3..0]의 counter가 4'b1111에 도착하면 TC 가 High가 된다. 이 핀을 MSB 쪽 [7..4]의 Counter의 CEP, CET에 연결하였다. 따라서 8'b00001111 이후 8'b00010000이되고 이후 schematic 상 앞 counter의 CEP, CET가 High 이면 8'b0001(hold)0001(count) 식으로 8bit counter가 동작하는 것을 확인 할 수 있다.

(버스 표현의 간단함을 위해 Hex code로 나타내었다.)

A picture containing screenshot

Description automatically generatedA picture containing screenshot

Description automatically generatedA picture containing screenshot

Description automatically generated

위와 마찬가지로 로드 기능 포함 카운터의 동작이제대로 출력 되는 것을 LED를 통해 확인 할 수 있다.