**디지털회로실험 보고서**

-미니 프로젝트 1-

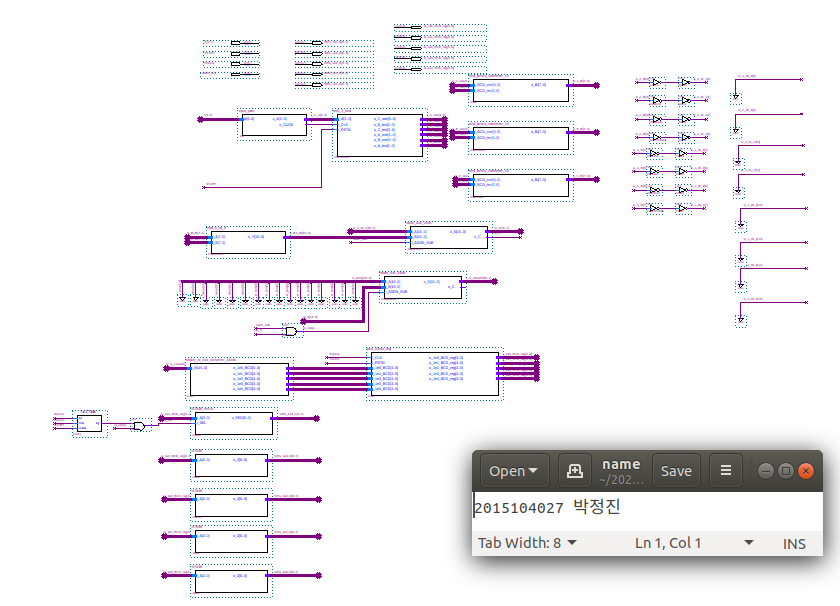
전자공학과

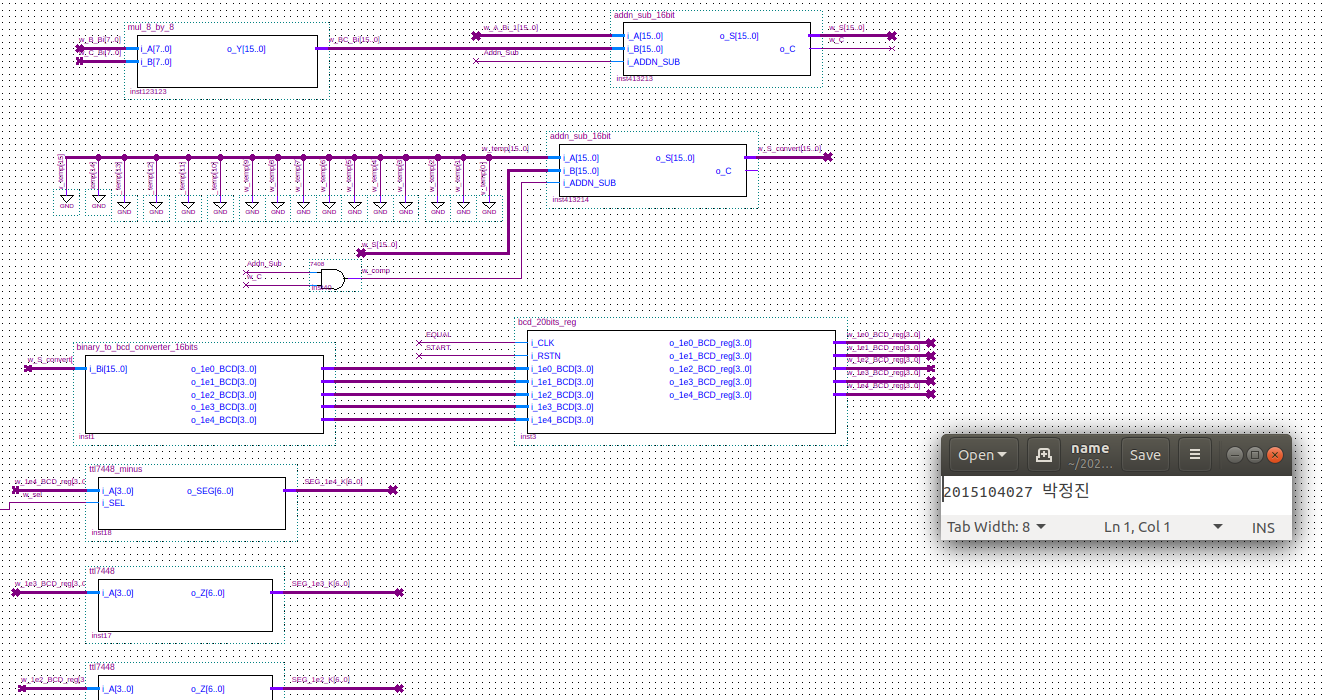
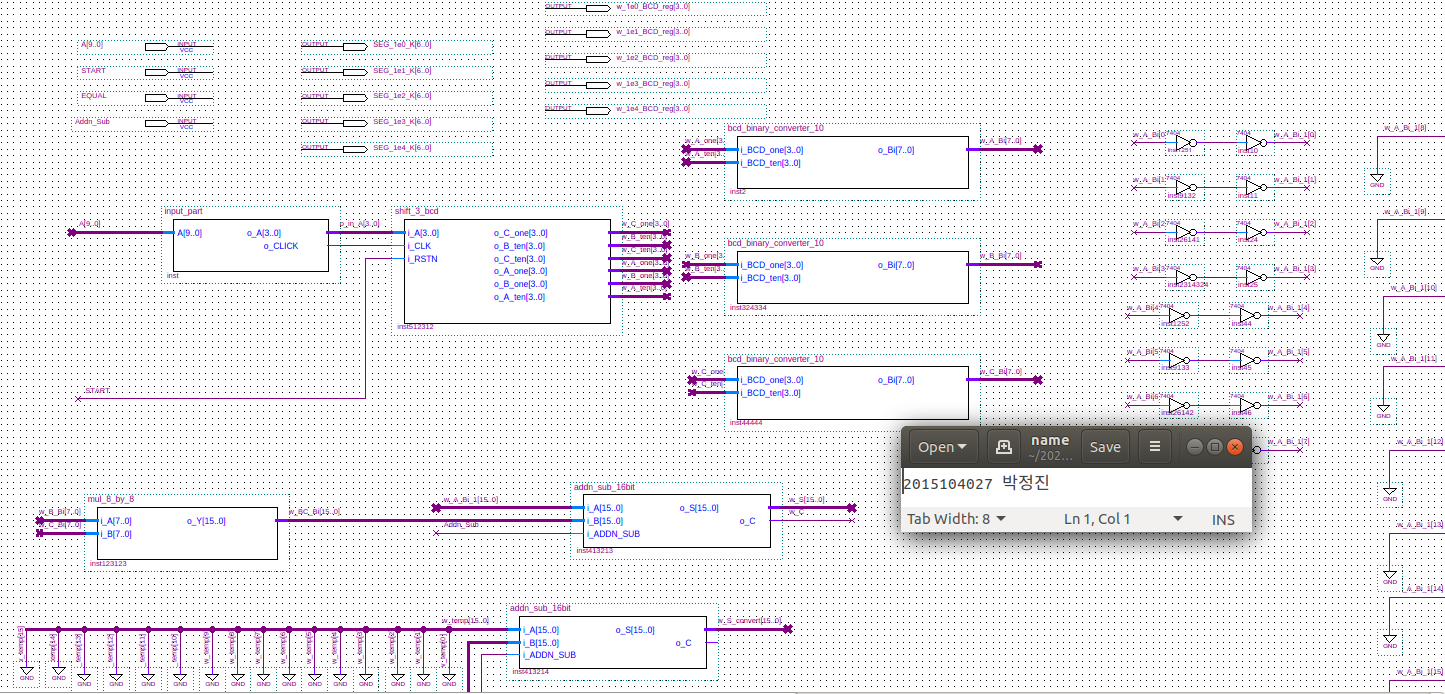
2015104027

박정진

**실험 결과**

Quartus Simulation (top.bdf)





기본적인 half adder부터 full adder, Ripple Carry Adder(이하 RCA), Multiply 연산을 하는 회로들은 hierarchy를 이용하여 bottom up design을 하였다. 74 시리즈의 primitive gate만을 이용해서 만들었고, 회로의 동작은 매우 정확함을 확인했다. 또한 Multiplier는 흔히 알고 있는 Series Multiply, 즉 Multiplicand를 Multiply 만큼 반복해서 더하며 accumulator에 쌓아가는 방식이 아닌 vedic multiply 로 설계를 하였다. Vedic multiply는 최소한의 and 게이트와 RCA를 이용하여 회로의 area가 줄어들 뿐만 아니라 accumulator에 쌓아가며 계산하는 것이 아니므로 계산을 하기위한 Delay가 단축 될 수 있다는 장점이 있다. Multiplier도 Adder와 마찬가지로 2x2 multiply 부터 4x4, 8x8 까지 순차적으로 hierarchy를 이용하여 설계를 진행하였다.

이번 회로가 저번 7주차 및 중간고사의 설계와 다른 점은 Symbol과 hierarchy를 이용했다는 점이다. Symbol을 이용한 회로 설계는 설계의 간편함과 특정 동작을 하는 회로를 원할 때 재사용이 가능하다는 점이 이번 설계를 용이하게 만든 중요한 포인트였다.

하나 설계에서 아쉬운 점은 Delay를 고려했을 때 16bit도 물론 적다면 적을 수 있는 비트 수 이지만 carry propagation delay로 인해 CSA(Carry Save Adder), CSKA(Carry Skip Adder), Hybrid Adder (CSA + CSKA) 에 비해 계산 속도가 느리다는 점이다. 하지만 시뮬레이션 상에서는 RTL Level의 회로만 동작 검증을 했으므로 Gate Level에서도 테스트를 한번 진행해야할 필요성을 느낀다.

또한 EQUAL 버튼을 누르기 전에는 눌렀던 숫자가 표시되지 않는 점이 있어 회로의 중간에 관련 기능을 할 수 있도록 MUX를 넣어 Shift register에서 나오는 Output 과 연결하는 작업을 해야 할 것 같다.

동작 원리 :

동작 원리는 이전과 같다. START 버튼으로 초기화를 시키고, 자기가 원하는 숫자를 6개를 누른다. 그러면 계산식이 Output = (Ax10+Ax1) +/- (Bx10+Bx1) \* (Cx10+Cx1) 의 형태로 계산 되게 하였다. 그리고 숫자 6개를 누른 후 EQUAL 버튼을 눌러야만 Segment에 결과가 나오고 새로운 계산을 시작할 땐 다시 START를 눌러 초기화를 시키면 된다.

회로의 구성

Shift Register

BCD

to

Binary\_Converter

Input Part

20bits

Parallel

Register

Binary

to

BCD\_Converter

Addn\_Sub

&

Multiply

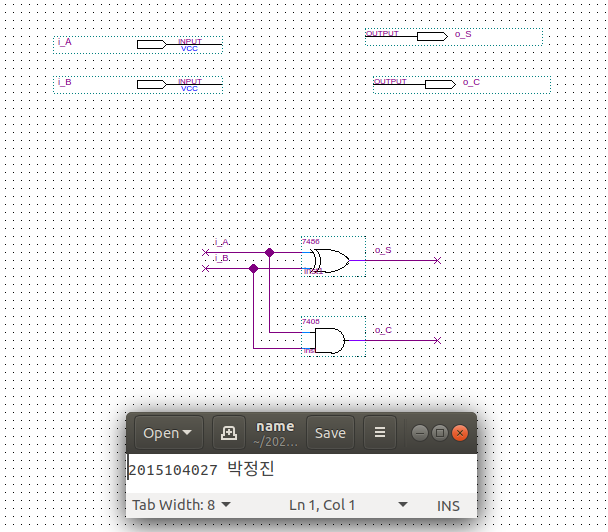
ttl7448

&

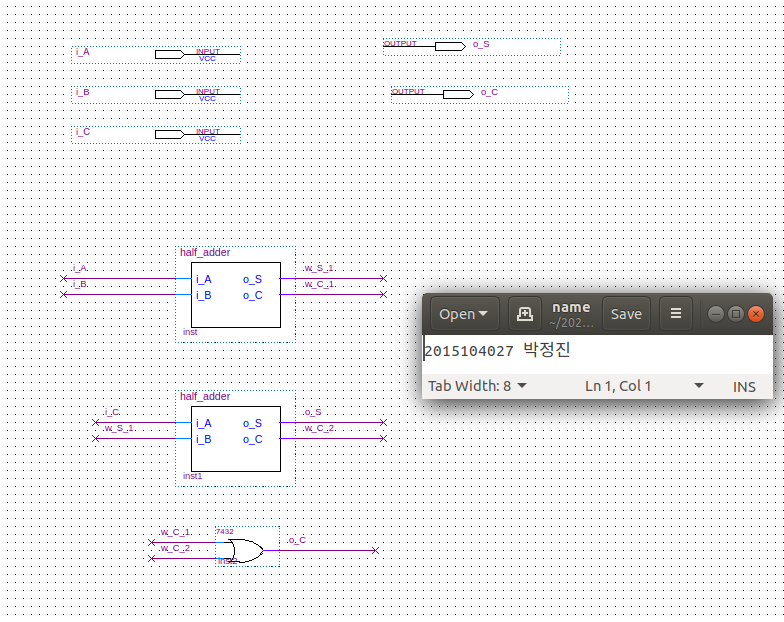
ttl7448\_minus

Sub-Module Schematic

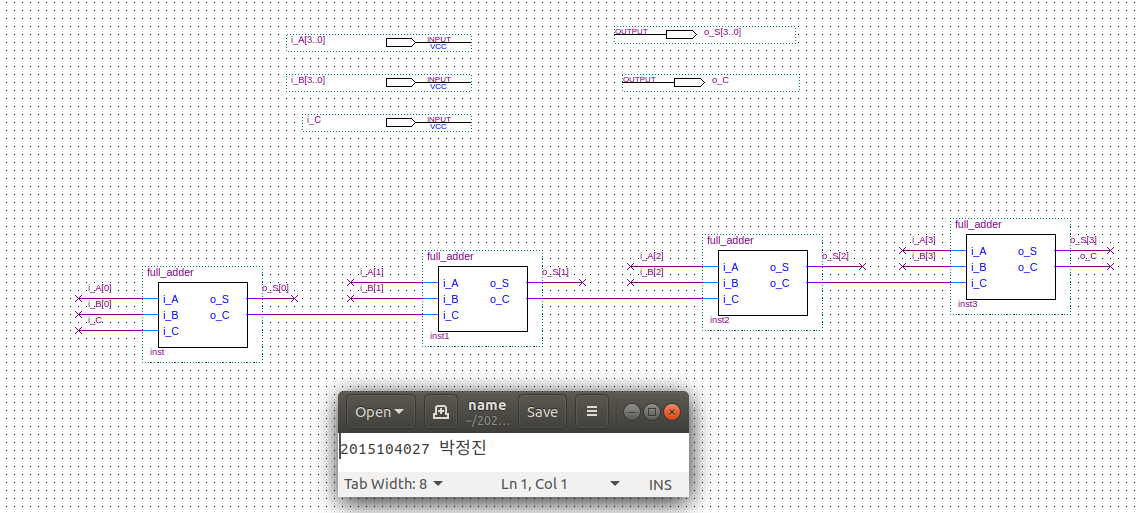
half adder



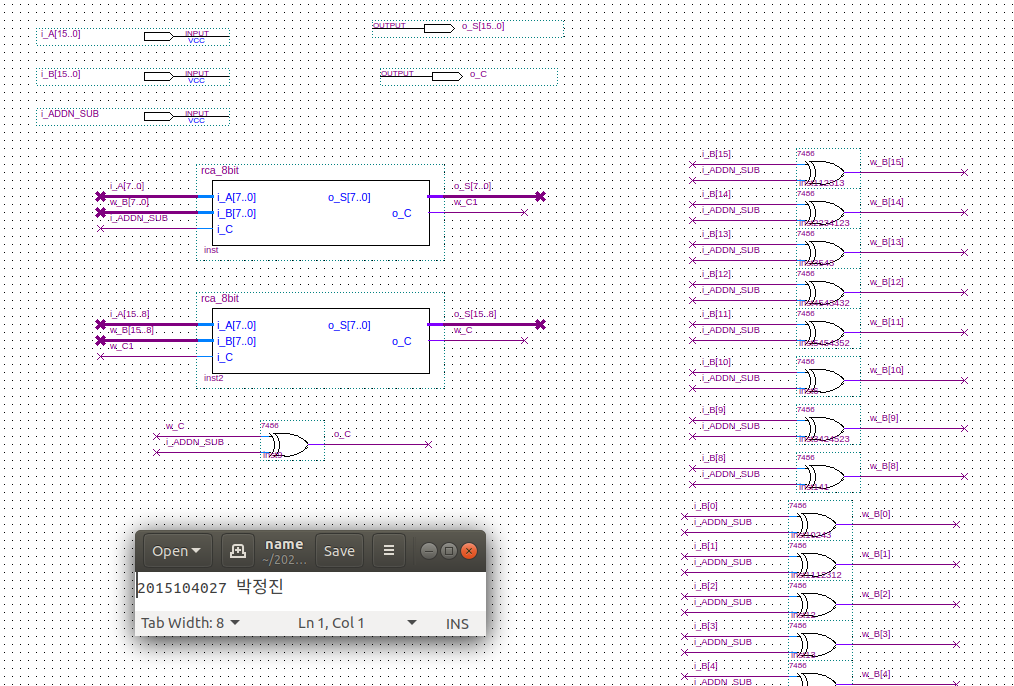
full adder



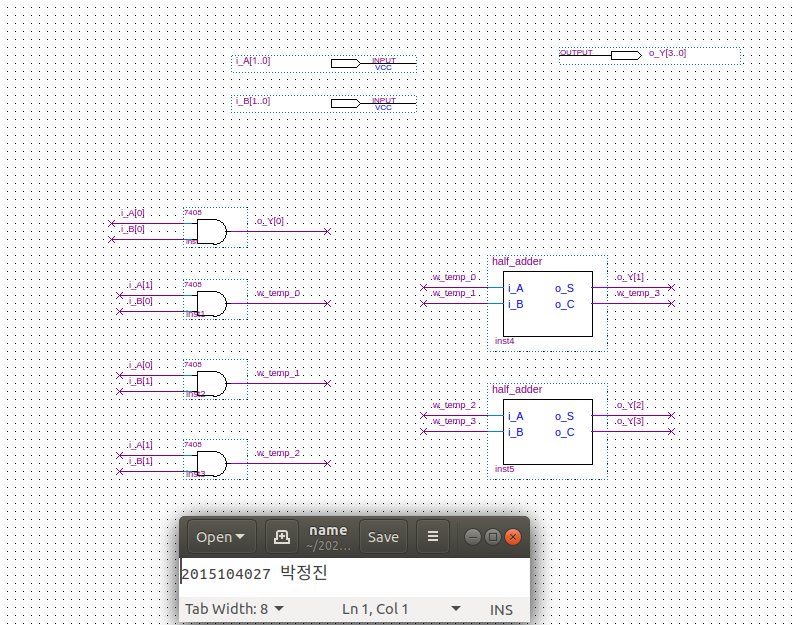
RCA 4bits



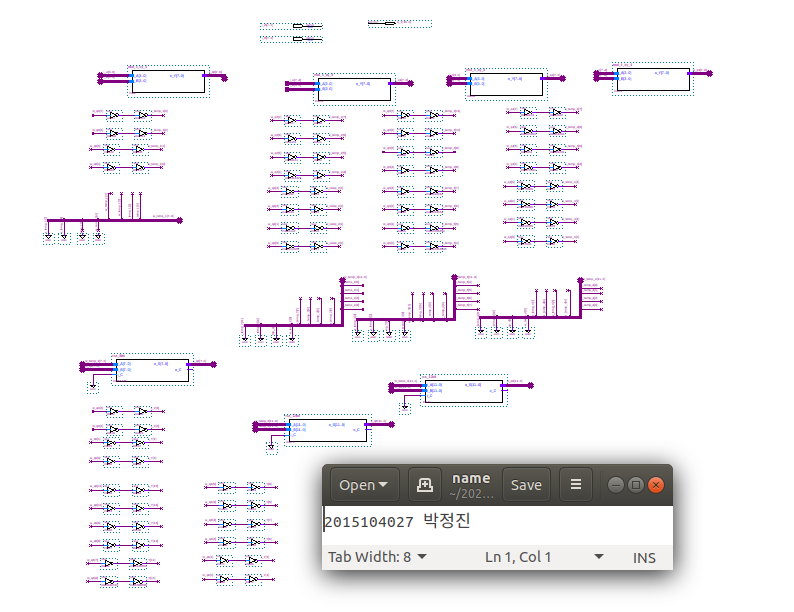
Addn\_Sub\_16bits

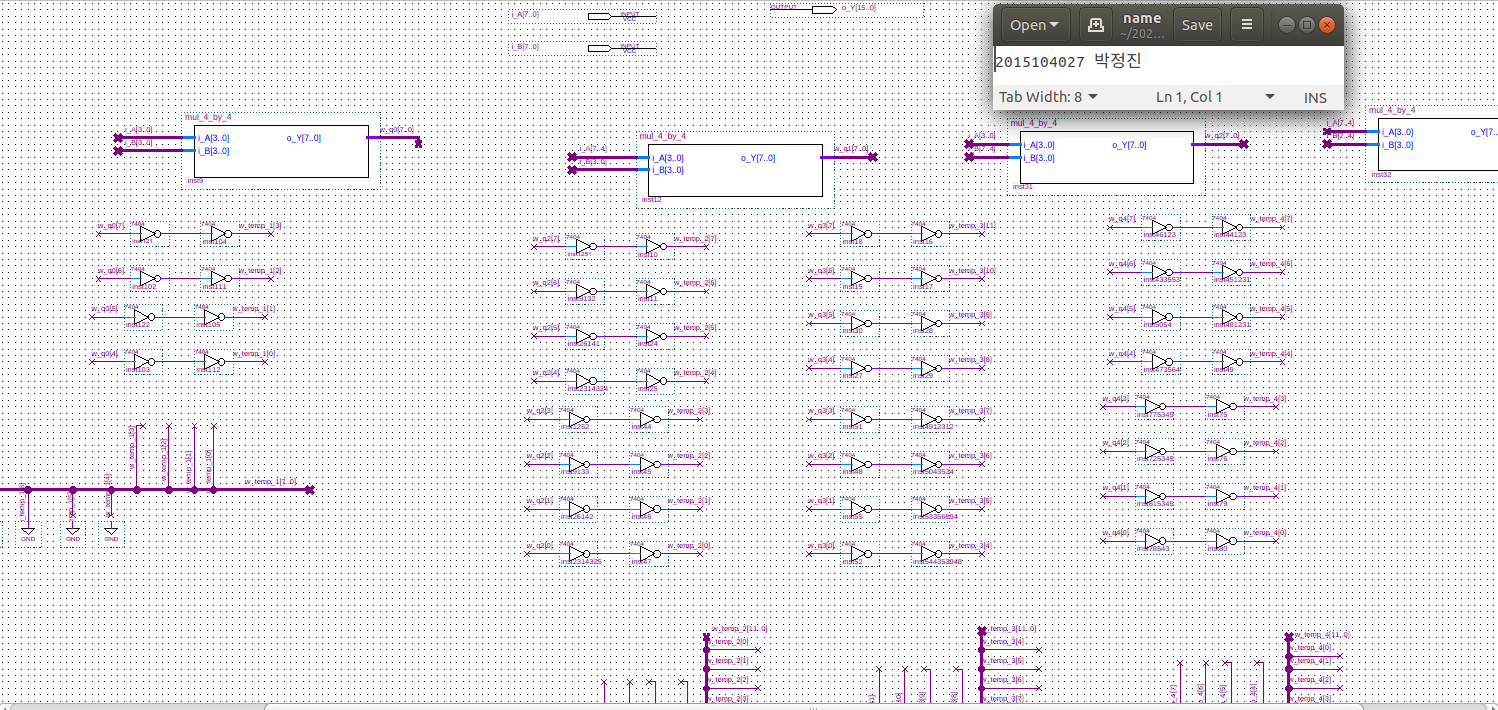


Vedic\_Multiplier\_2\_by\_2

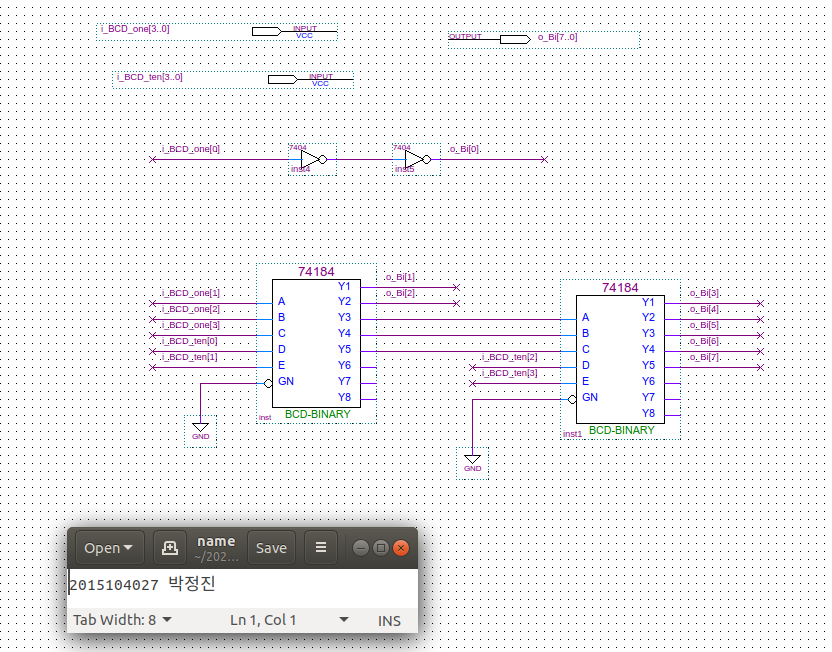


Vedic\_Multiplier\_8\_by\_8

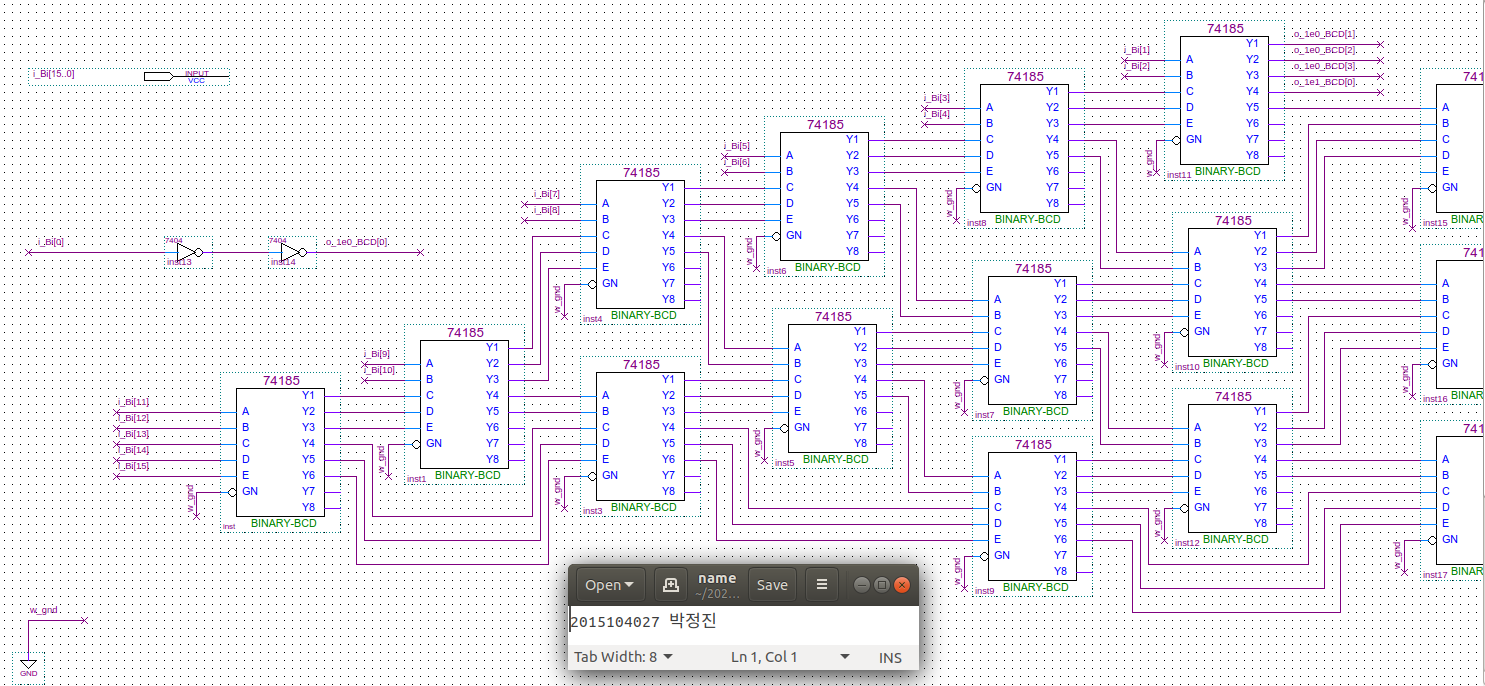
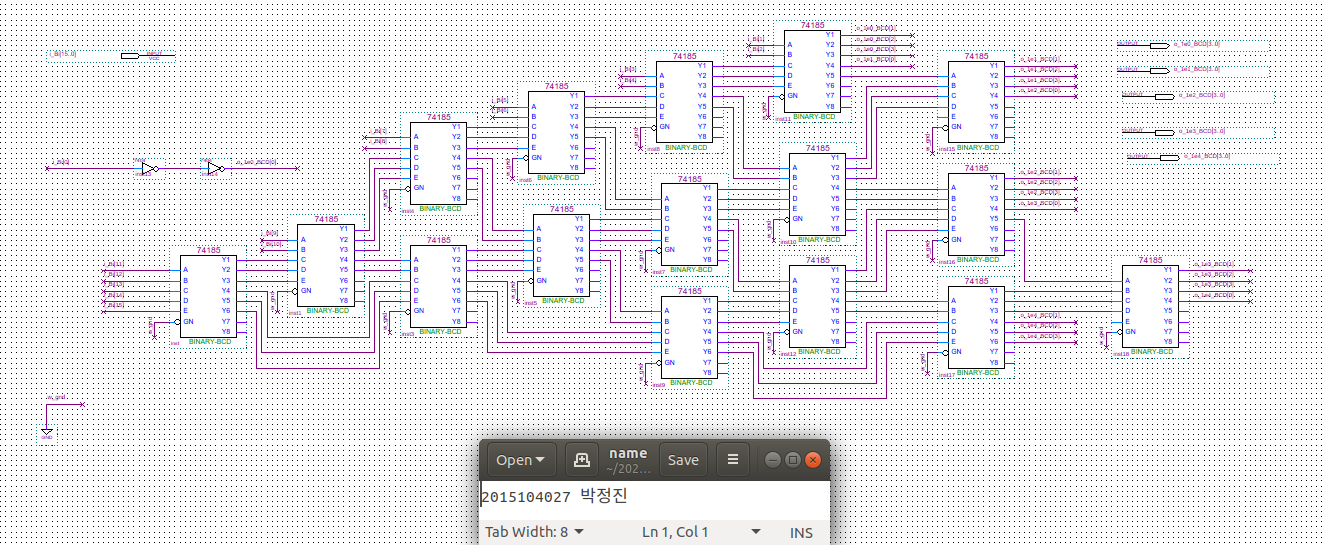




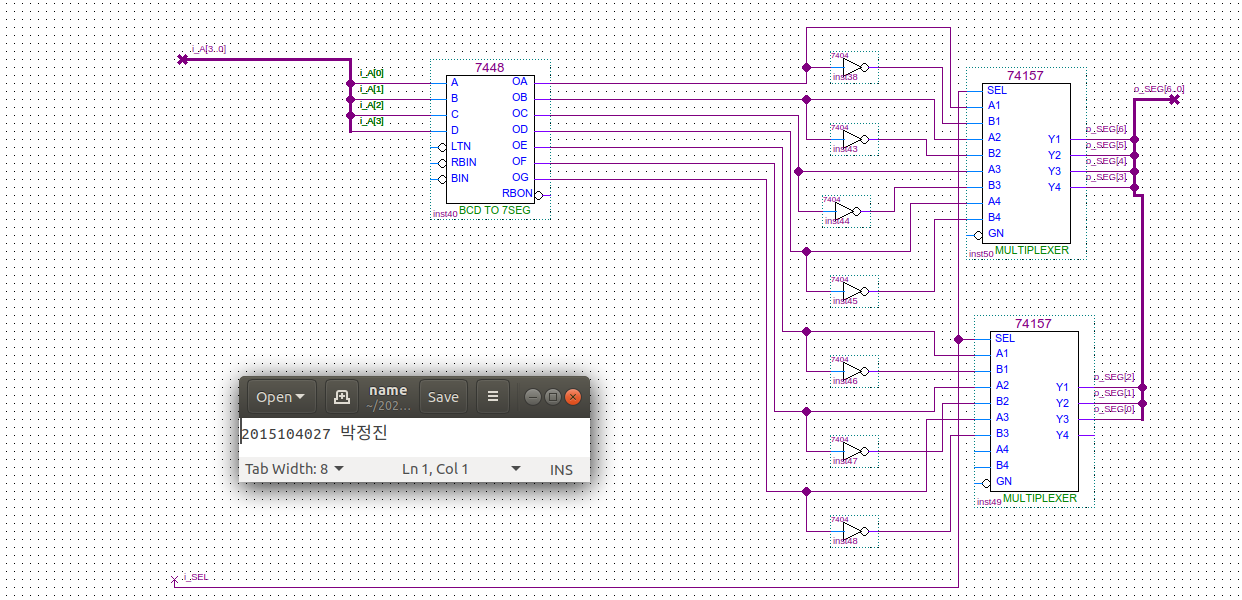
BCD\_to\_Binary\_Converter



Binary\_to\_BCD\_Converter



ttl7448\_minus



Simulation

Case 1)

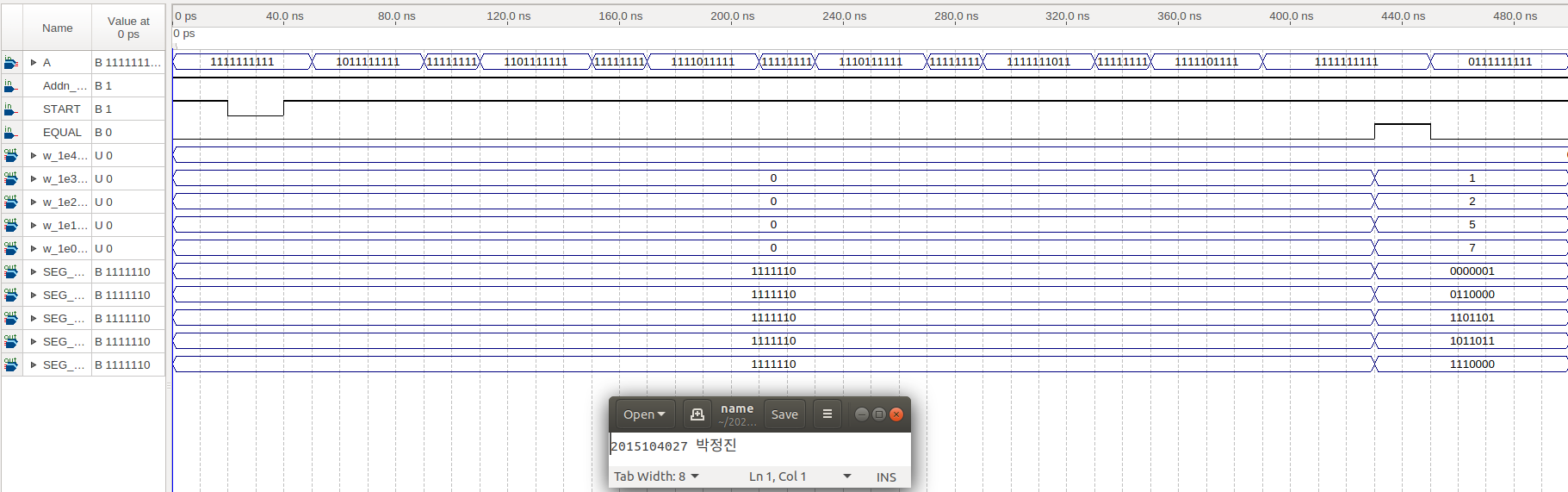
A = 87, B = 56, C= 24

Addn\_Sub = 1'b1 (87-56x24)

결과 예상 : -1257

EQUAL rising(0->1) 과 함께 계산 결과가 segment에 제대로 표시 되는 것을 확인 할 수 있다.

BCD 결과가 잘 나오는 것을 확인 할 수 있고, 또한 부호를 나타내는 맨 위 segment가 g만 켜진 상태, 즉 음수 부호를 정확히 표시하는 것을 볼 수 있다. (segment : -1257)

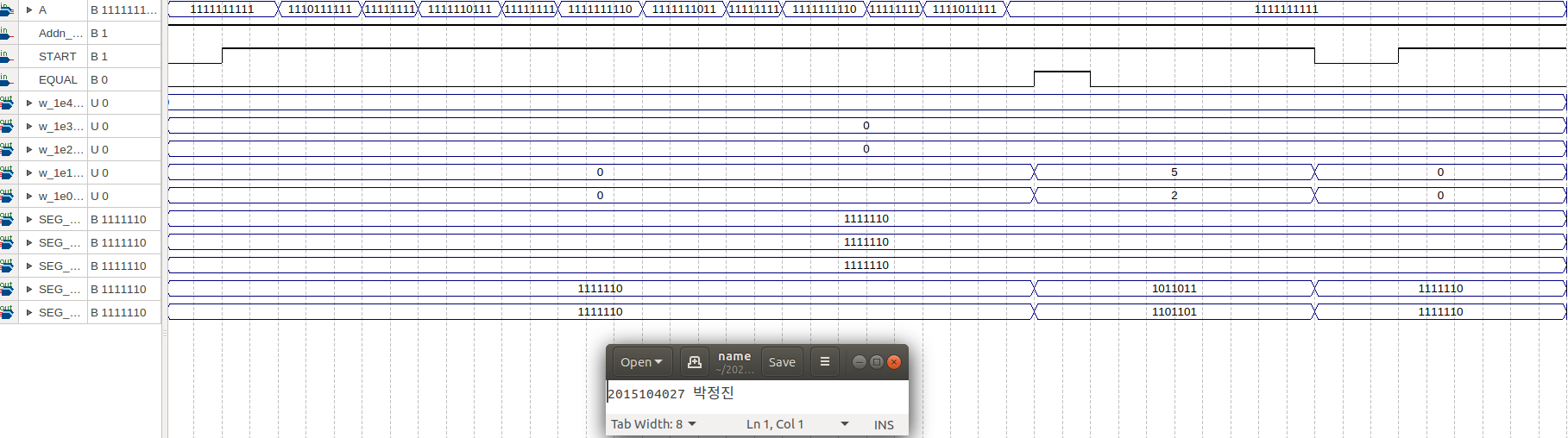


Case 2)

A = 62, B = 02, C= 05

Addn\_Sub = 1'b1 (62-2x5)

결과 예상 : 00052



BCD코드도 잘 나오며, 부호도 0으로 양수를 표시하는 것을 확인 할 수 있다. (segment : 00052)

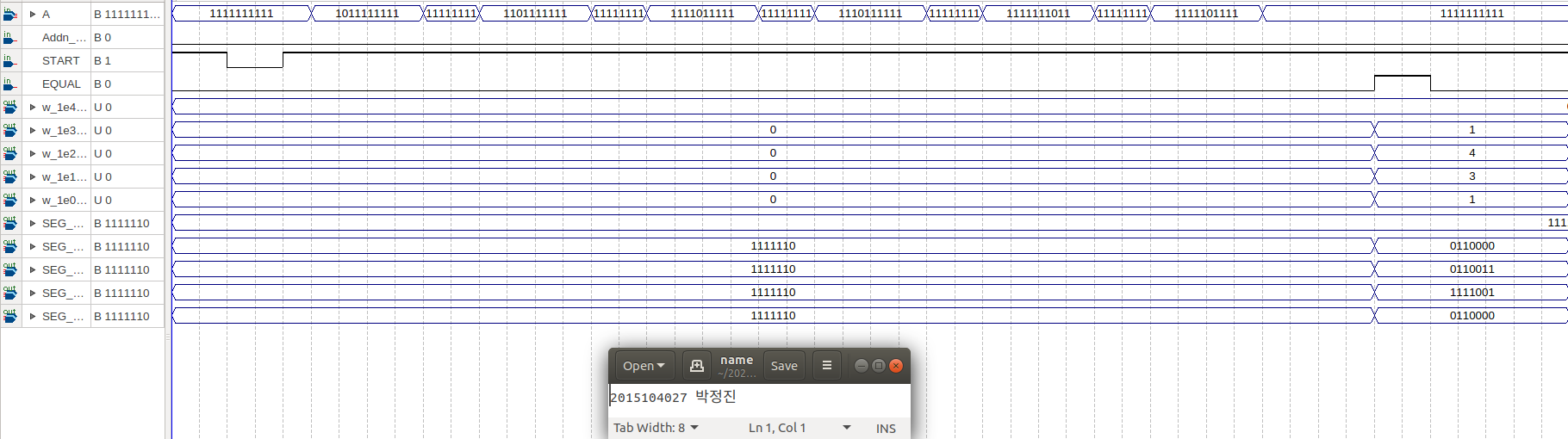
Case 3)

A = 87, B = 56, C= 24

Addn\_Sub = 1'b0 (87+56x24)

결과 예상 : 01431

BCD코드가 잘 나오며, 부호도 0으로 양수를 표시하는 것을 확인 할 수 있다. (segment : 01431)



Case 4)

A = 60, B = 09, C= 05

Addn\_Sub = 1'b0 (60+09x05)

결과 예상 : 00105

BCD코드가 잘 나오며, 부호도 0으로 양수를 표시하는 것을 확인 할 수 있다. (segment : 00105)

