디지털시스템설계 Lab4

A14 20230227 강현우 20230024 문요준

1. 개요

이진수 덧셈에 사용하는 반가산기와 전가산기의 기능을 이해하고 회로를 구성한 후, 이를 사용해 덧셈, 뺄셈, 곱셈 회로를 구성한다. 자세한 학습 목표는 다음과 같다.

- 반가산기와 전가산기 구현
- 5-bit 리플 가산기/감산기 구현
- 5x3 이진 곱셈기 구현
- 테스트 벤치를 이용한 회로 검증

2. 이론적 배경

반가산기는 1-bit 이진수 두 개를 입력받아 합과 Carry out을 출력하는 회로이다. 전가산기는 1-bit 이진수 두 개와 이전 가산기의 Carry in을 입력받아 합과 Carry out을 출력하는 회로로, 두 개의 반가산기를 연결해 구현할 수 있다.

N-bit 가산기는 N-bit 이진수 두 개를 더하여 N-bit 덧셈 결과와 Carry out을 출력하는 가산기이다. N-bit 가산기는 다양한 방법으로 구현할 수 있는데, 대표적인 예시로 N-bit 리플 가산기가 있다. N-bit 리플 가산기는 N개의 전가산기를 순차적으로 이어 구현하며, 각 전가산기가 각 자릿수의 연산을 맡는다. 이때 k번째 자릿수를 계산하는 전가산기의 Carry out이 k+1번째 자릿수를 담당하는 전가산기의 Carry in에 연결되어 가장 낮은 자릿수부터 가장 높은 자릿수까지 Carry가 순차적으로 전파된다. 이러한 순차적인 연결 구조로 인해 낮은 자릿수의 연산이 끝나 Carry out이 결정되어야 높은 자릿수의 연산을 시작할 수 있어서 자릿수가 많아질수록 속도가 매우 느려진다. 한편, A-B 꼴의 뺄셈은 2의 보수 성질을 활용해 A+(-B) 꼴의 덧셈으로 감산기를 간단히 구현할 수 있다.

MxN 이진 곱셈은 M-bit Multiplicand와 N-bit Multiplier의 각 자릿수의 부분 곱을 통해 얻은 이진수 N개를 합하여 계산한다. 부분 곱은 AND 연산으로, 부분 곱들의 덧셈은 가산기로 계산해 최종 결과을 구한다.

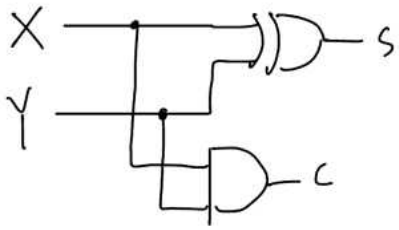
3. 실험 준비

1) 반가산기의 진리표와 식을 구하고 회로를 그린다.

X	Y	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

$S=X'Y+XY'=X\oplus Y$

$C=XY$

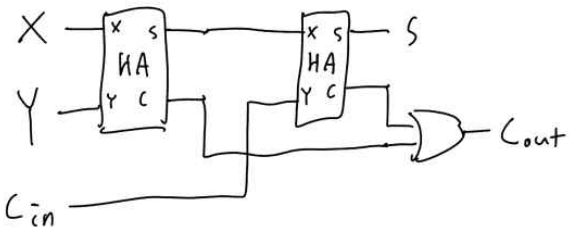


2) 전가산기의 진리표와 식을 구하고 반가산기를 이용해 전가산기 회로를 그린다.

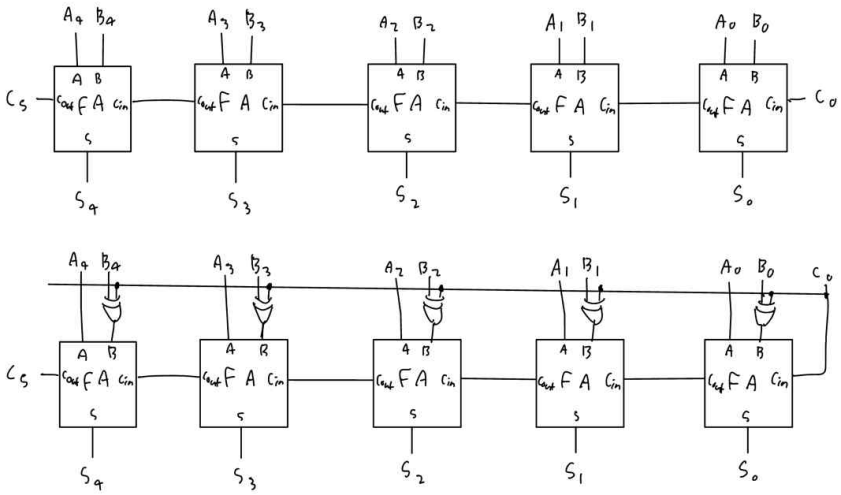
X	Y	Cin	S	Cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

S=X'Y'Cin+X'YCin'+XY'Cin'+XYCin=X⊕Y⊕Cin

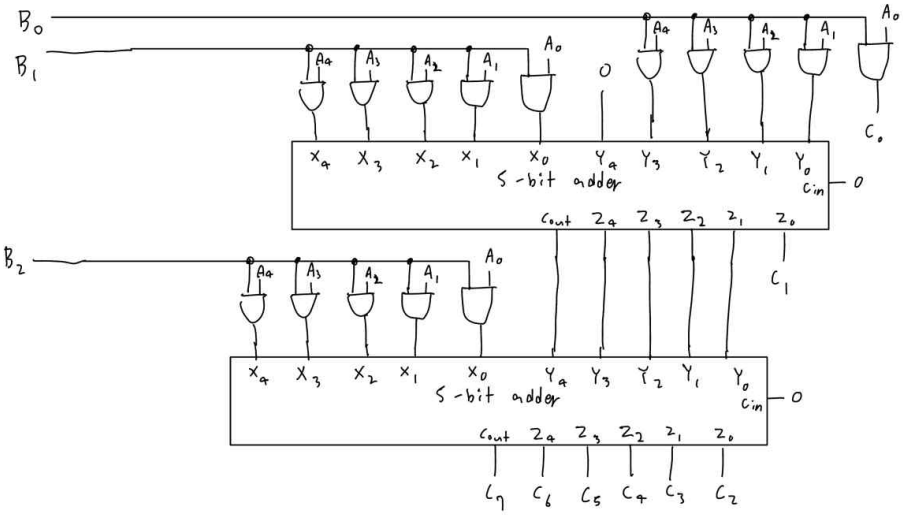
Cout=XY+XCin+YCin=XY+Cin(X⊕Y)



3) 전가산기를 사용해 5비트 리플 가산기와 감산기 회로를 그린다.(음수는 2의 보수로 표현한다.)

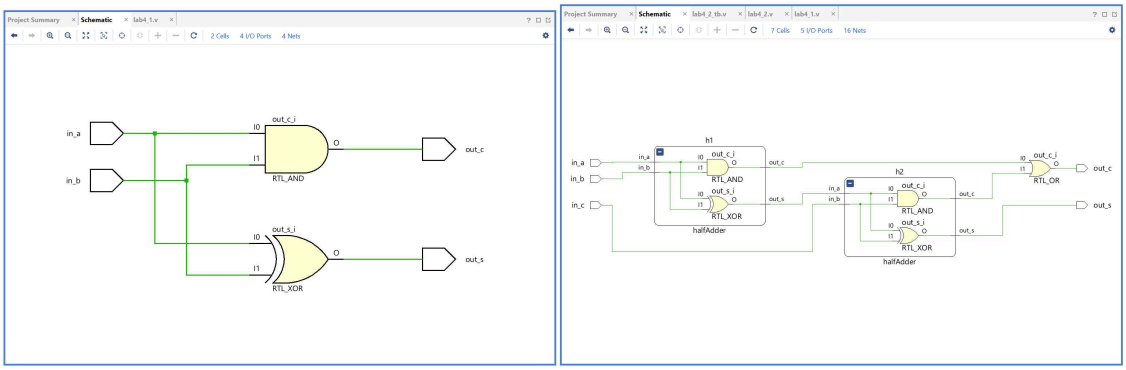


4) 5비트 리플 가산기를 사용해 5x3 이진 곱셈기 회로를 그린다.



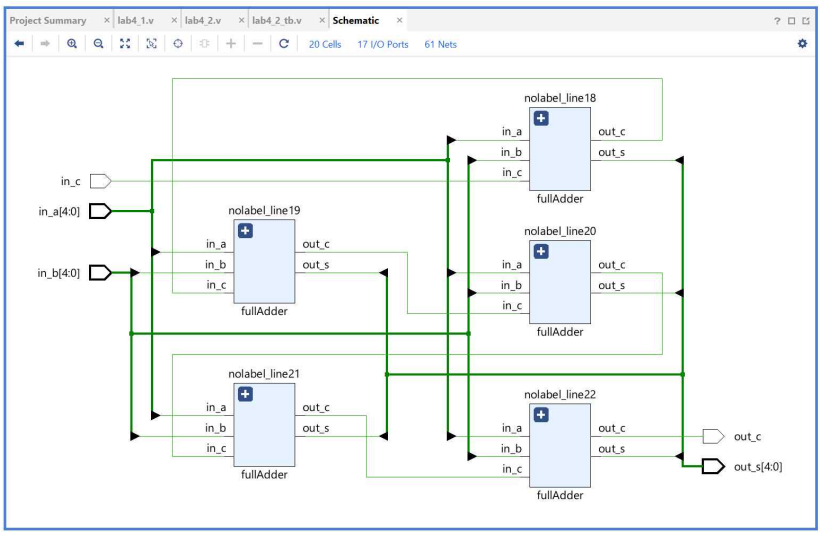
4. 결과

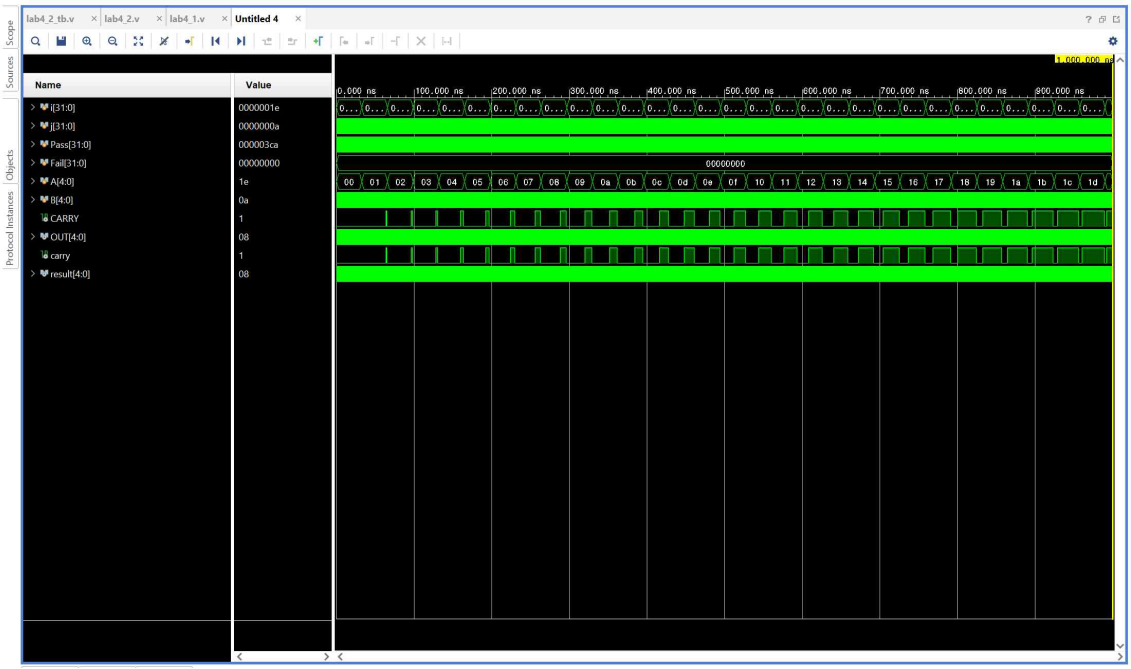
i) lab4_1.v



반가산기 회로와 반가산기로 구현한 전가산기 회로가 잘 구현된 것을 확인할 수 있다.

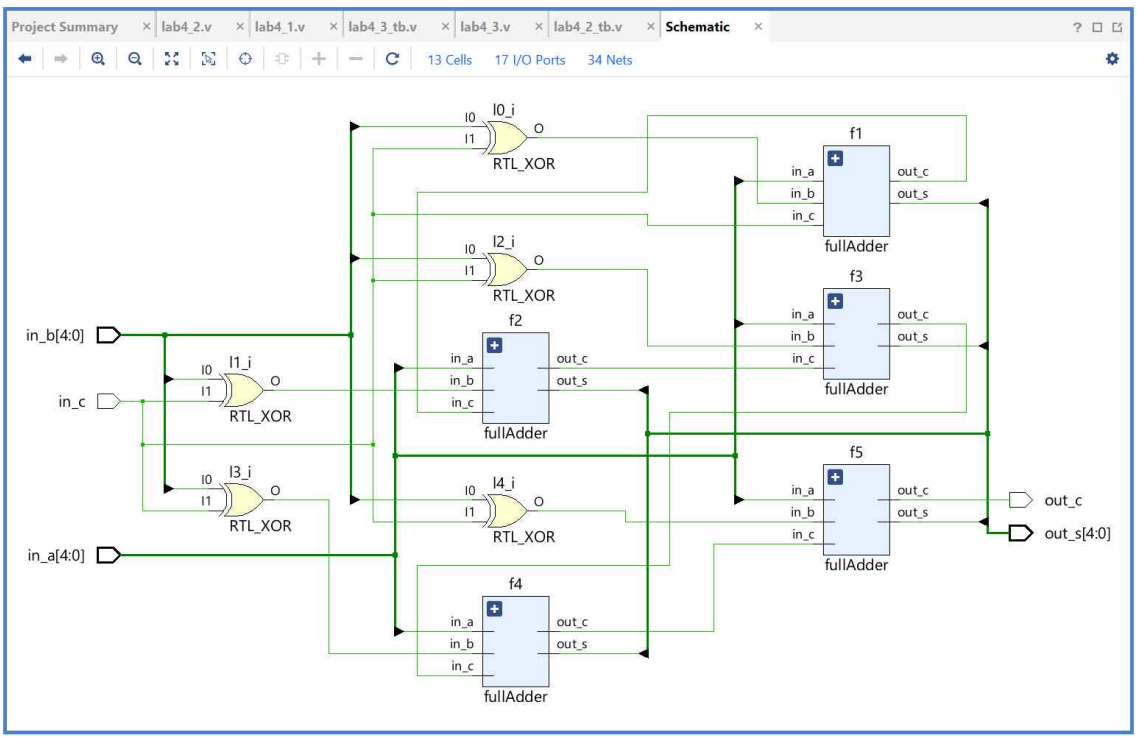
ii) lab4_2.v

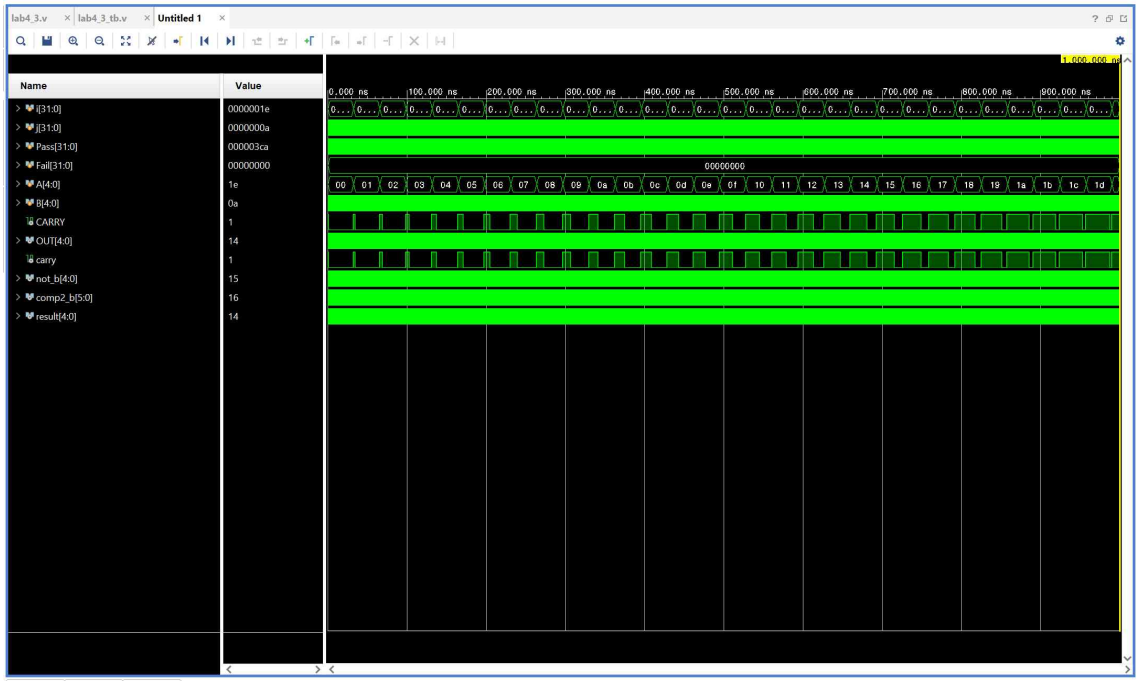




전가산기를 이용해 구현한 5-bit 리플 가산기의 회로가 잘 구현된 것을 확인할 수 있다. 또한, 테스트벤치를 이용해 회로를 시험해본 결과, 회로가 실제 결과와 맞지 않는 부분이 있을 때마다 값이 1씩 증가하는 변수인 fail 변수가 0으로 출력되었다. 따라서 5-bit 리플 가산기가 잘 구현된 것을 확인할 수 있다.

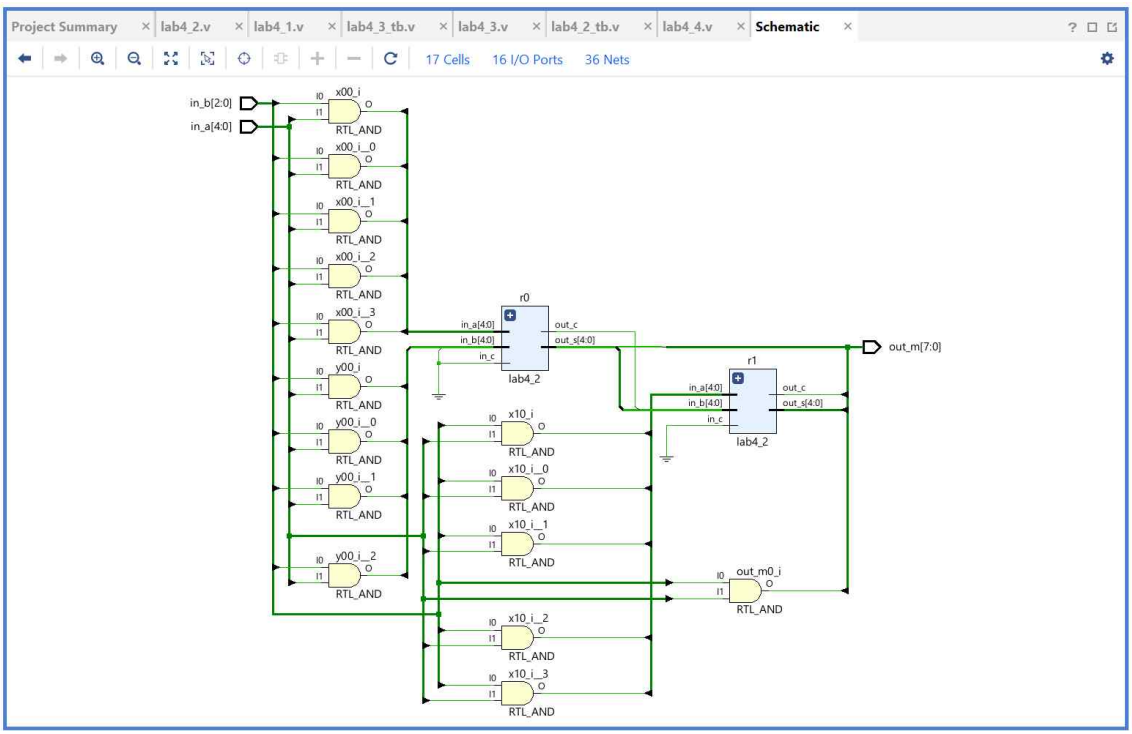
iii) lab4_3.v

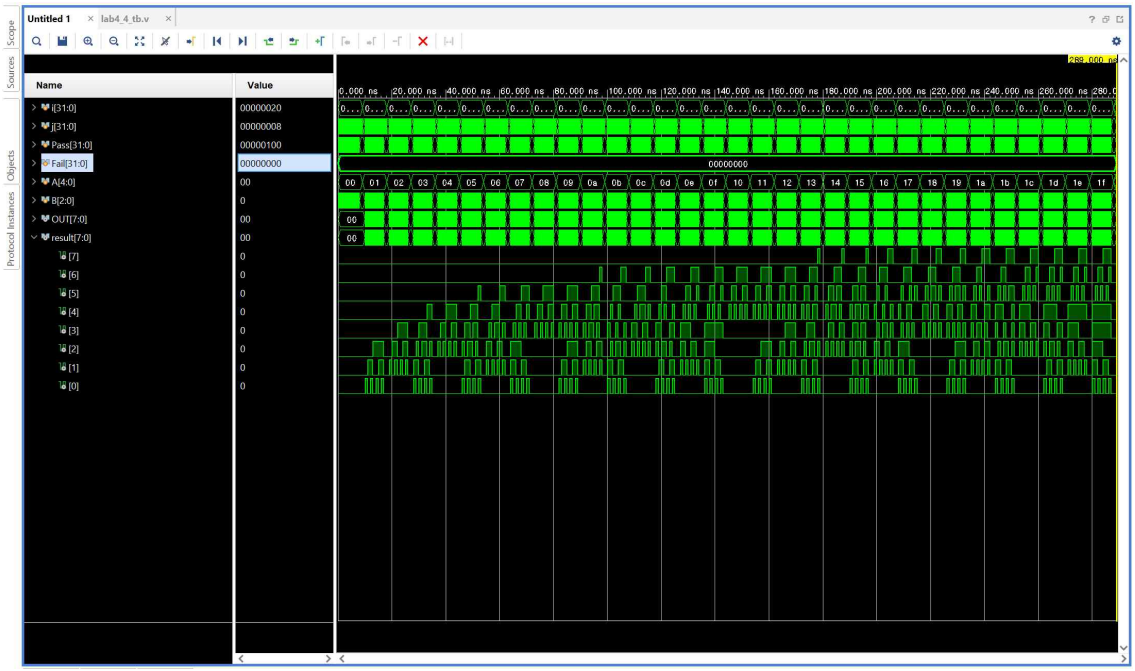




전가산기를 이용해 구현한 5-bit 리플 감산기의 회로가 잘 구현된 것을 확인할 수 있다. 또한, 테스트벤치를 이용해 회로를 시험해본 결과, 회로가 실제 결과와 맞지 않는 부분이 있을 때마다 값이 1씩 증가하는 변수인 fail 변수가 0으로 출력되었다. 따라서 5-bit 리플 감산기가 잘 구현된 것을 확인할 수 있다.

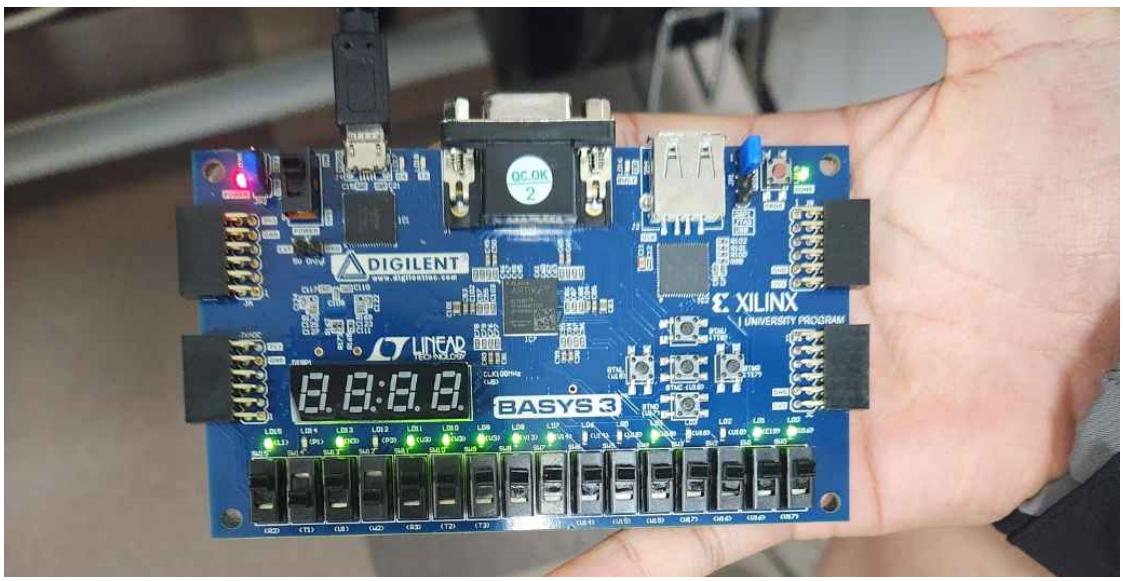
iv) lab4_4.v





5비트 리플 가산기를 사용해 구현한 5x3 이진 곱셈기의 회로가 잘 구현된 것을 확인할 수 있다. 또한, 테스트벤치를 이용해 회로를 시험해본 결과, 회로가 실제 결과와 맞지 않는 부분이 있을 때마다 값이 1씩 증가하는 변수인 fail 변수가 0으로 출력되었다. 따라서 5x3 이진 곱셈기가 잘 구현된 것을 확인할 수 있다.

v) lab4_fpga.v



lab4_fpga.v 파일을 이용해 5x3 곱셈기를 회로에 구현한 결과 10101 × 111 = 10010011 연산이 잘 수행된 것을 확인할 수 있었다.

5. 논의

반가산기와 전가산기에서 시작해서 n-bit 리플 가/감산기와 곱셈기까지 사실상 모든 기본 연

산을 논리 회로만 이용해서 구현하면서 컴퓨터의 근본적인 작동 체계에 대해 더 자세히 알 수 있었다. 이번 과제를 하며 그동안 익힌 verilog에서 사용해본 모든 문법을 다 사용해야 했는데, 이를 통해 verilog 프로그래밍에 더 익숙해질 수 있었다. 또한, verilog의 schematic은 논리 게이트 수가 많아질수록 우리가 일반적으로 실험 준비 단계에서 그린 회로와 외적으로 너무 다르게 보이기 때문에 테스트벤치가 꼭 필요하다는 것을 느낄 수 있었다.