

# 디지털시스템설계 Lab5

20230227 강현우 20230024 문요준

## 1. 개요

컴퓨터의 기초가 되는 산술 논리 장치(Arithmetic Logic Unit; ALU)와 정보를 저장할 수 있는 JK 플립플롭(Flip-flop)을 구현한다.

## 2. 이론적 배경

ALU는 입력에 대해 여러 산술(Arithmetic) 및 논리(Logic) 연산을 수행한다. 연산의 종류에 따라 산술 장치와 논리 장치 두 부분으로 나눌 수 있는데, 산술 장치는 사칙 연산 등을, 그리고 논리 장치는 Bitwise 논리 연산 등을 맡는다.

회로는 비동기 회로와 동기 회로로 나눌 수 있는데, 클록을 따르지 않는 순차 회로와 모든 조합 회로는 비동기 회로다. 동기 회로는 다른 회로와 같은 순간에 맞춰 작동하기 위해 클록 신호를 따른다.

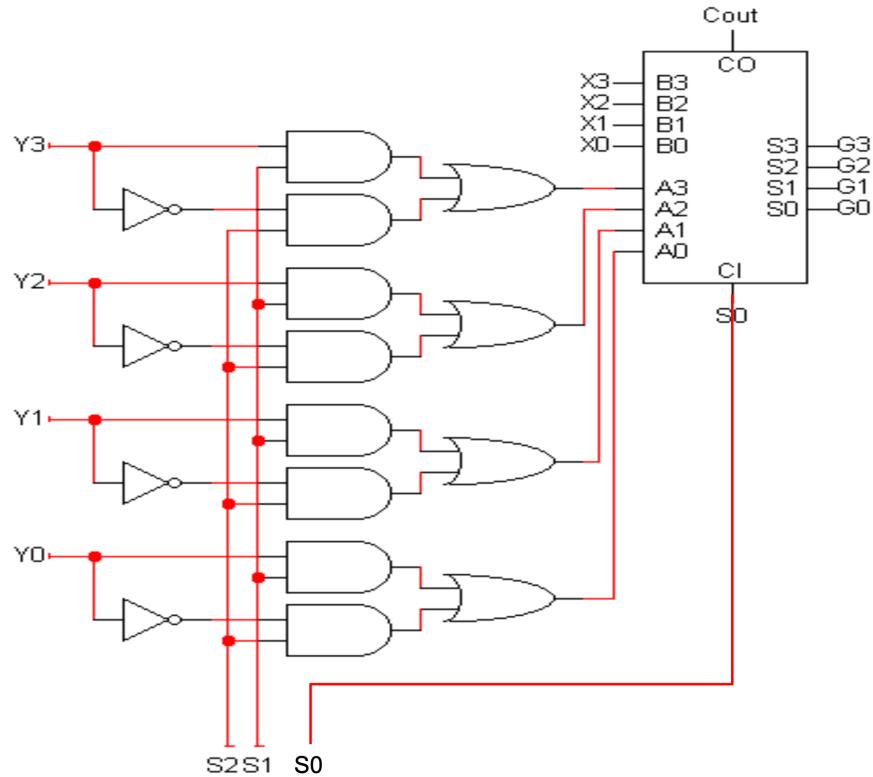
JK 래치는 SR 래치에 추가적인 회로를 더해 S와 R이 동시에 1인 상황에서도 정상적으로 작동하도록 수정한 것이다. JK 래치에서 J와 K가 동시에 1일 경우 현재 상태에 상관없이 값을 반전시킨다. 래치가 입력이 바뀔 때 출력도 바로 바뀌는 비동기 회로라면 플립플롭은 입력이 바뀌더라도 출력에 맞추어 반영되는 동기 회로이다. 즉 JK 플립플롭은 JK 래치가 클록 신호를 추가로 받아 이에 맞추어 작동하게 된 회로이다.

Master-slave JK 플립플롭은 SR 래치 두 개를 연결하여 만든 플립플롭으로, 클록이 1인 동안 Master 래치를 활성화해 입력을 임시로 저장한 뒤 클록이 0이 되는 순간 Slave 래치로 전달한다. 따라서 Master 래치가 활성화되어있는 동안 글리치로 잠깐 입력값이 생기면 이 값이 Master 래치에 저장되어있다가 다음 클록이 0이 되는 순간에 Slave 래치로 전파되는 문제가 생긴다. 이는 클록이 1인 동안 계속 입력을 받기 때문에 생기는 문제로 클록이 바뀌는 순간에만 입력을 받는 Edge-trigger 회로를 사용하여 해결할 수 있다.

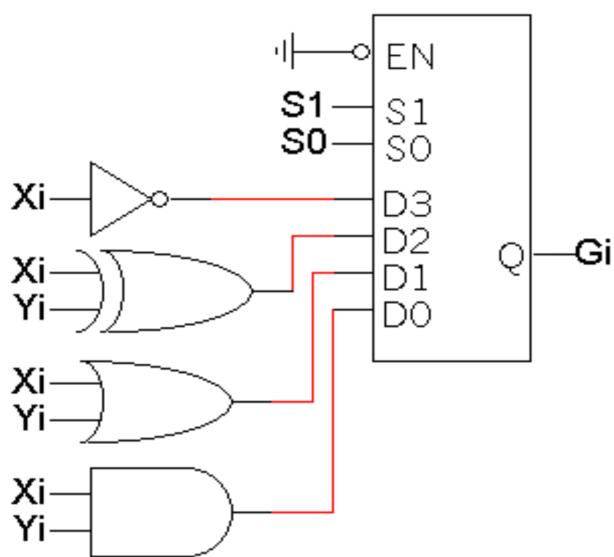
### 3. 실험 준비

1) 표 1의 ALU를 S3의 값에 따라 산술 장치와 논리 장치 두 부분으로 나누어 단순화하고 회로도를 그린다. 두 모듈을 2:1 MUX로 묶어 ALU의 회로도를 그린다.

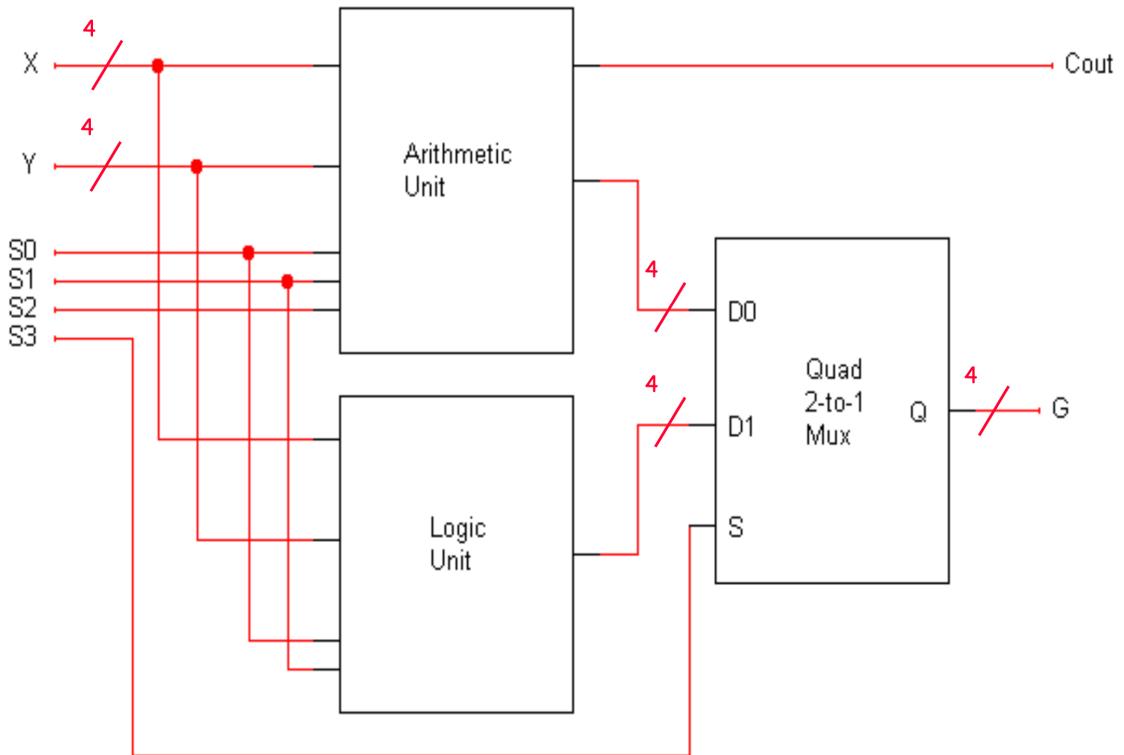
- Arithmetic Unit



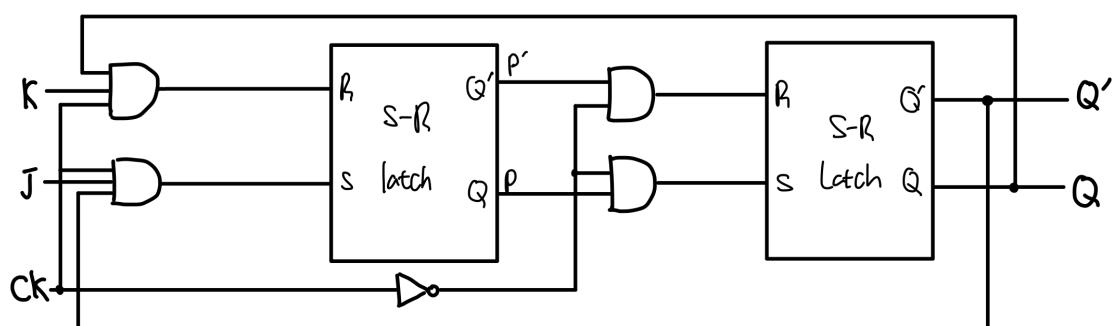
- Logic Unit



- ALU

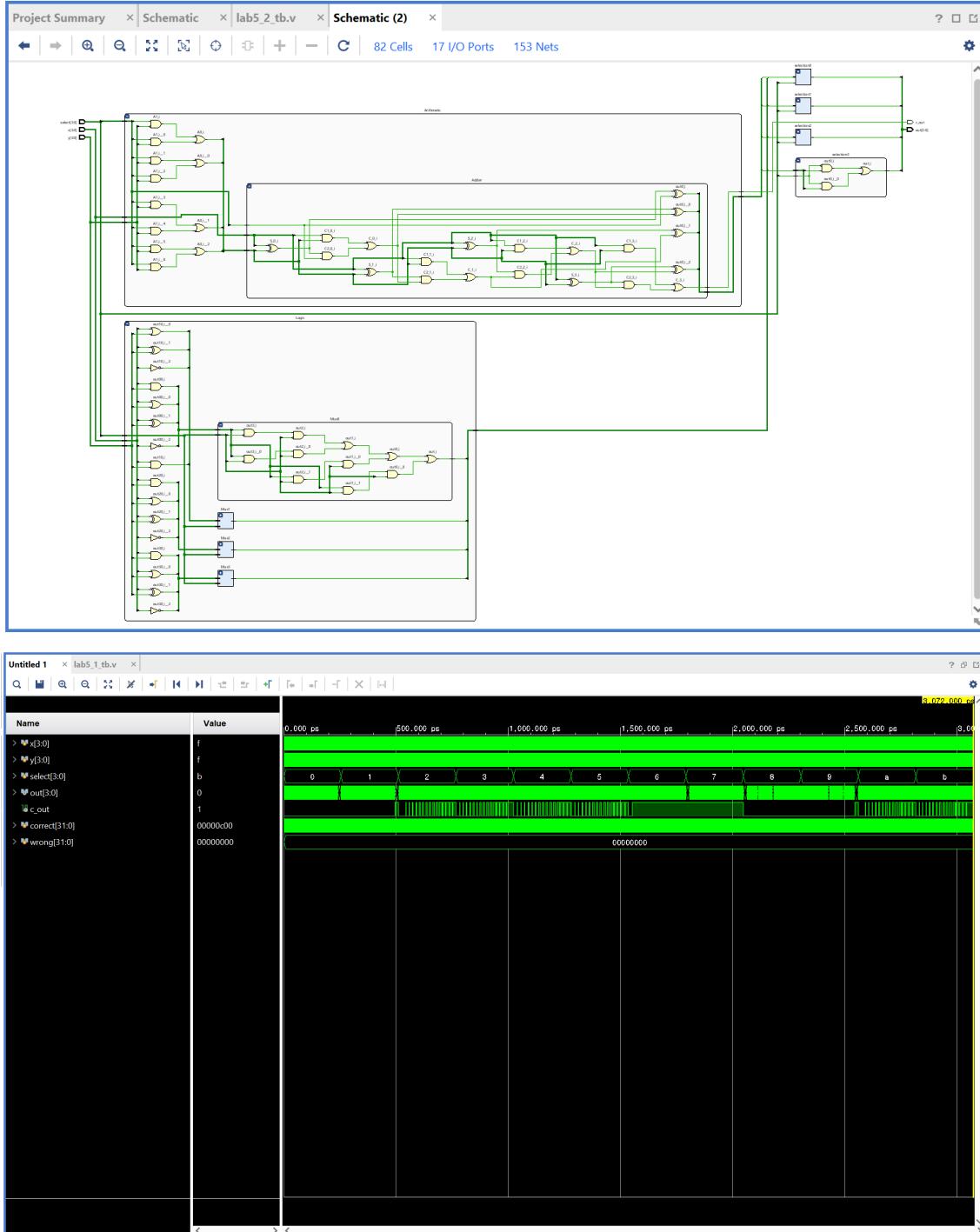


2) SR 래치의 회로도를 그린다. SR 래치를 사용해 Negative reset Master-slave JK 플립 플롭의 회로도를 그린다. SR 래치와 비교해 Master-slave JK 플립플롭이 해결 가능한 글리치와 해결할 수 없는 글리치를 예상하고 분석한다.



## 4. 결과

i) lab5\_1.v



가산기와 논리 회로, 그리고 2:1, 4:1 MUX를 이용해 구현한 ALU의 회로가 정상적으로 그려지는 것을 확인할 수 있었다. 또한, 테스트벤치 실행 결과 오류가 있을 때마다 1씩 증가하는 wrong 변수가 0으로 출력되는 것으로 ALU가 잘 구현된 것을 확인할 수 있었다.

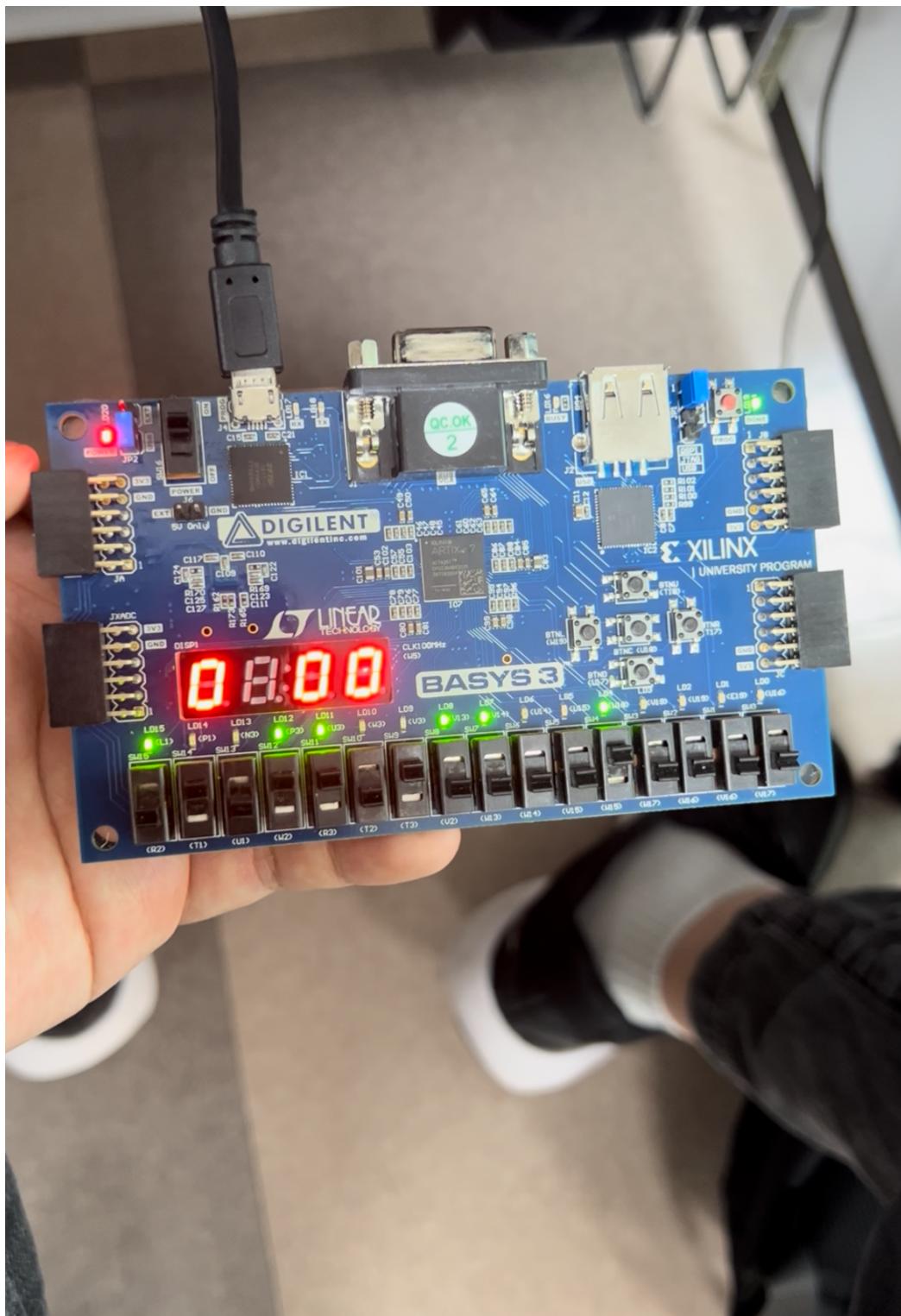
ii) lab5\_2.v



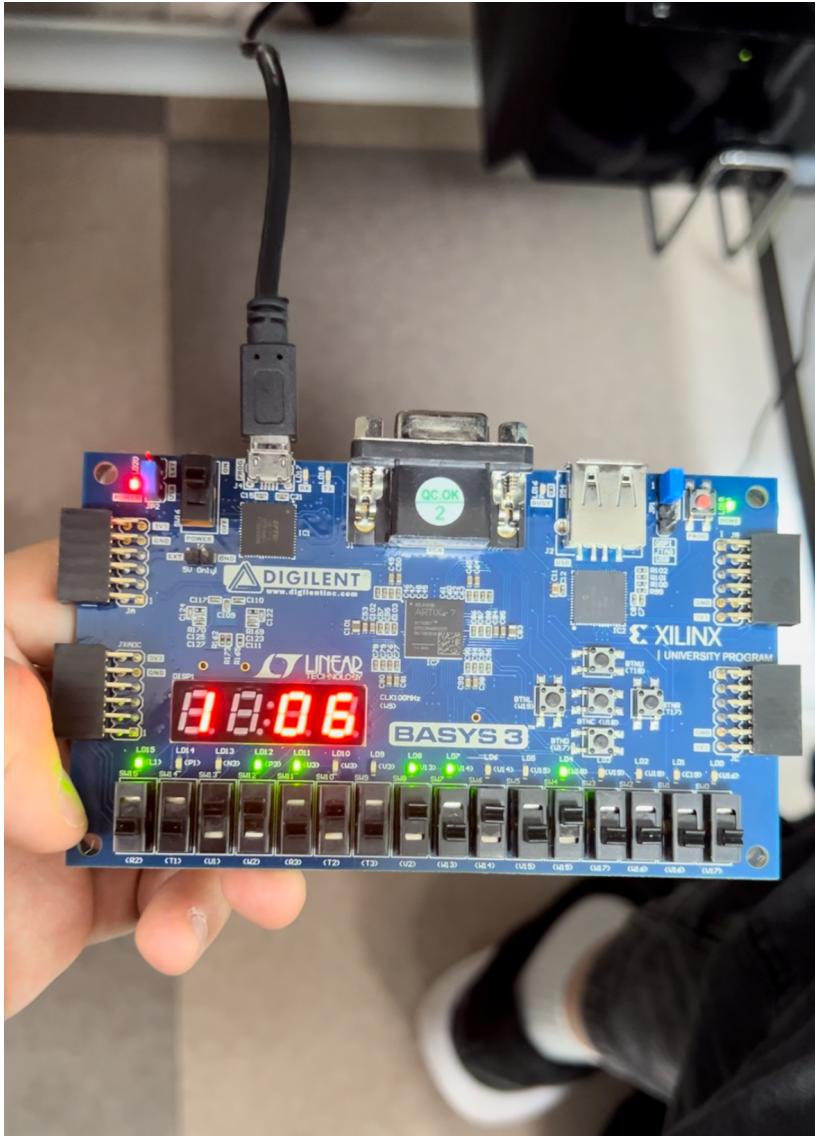
SR 래치 회로와 SR 래치로 만든 Master-Slave JK 플립플롭 회로가 잘 만들어지는 것을 확인할 수 있었다. 또한, 테스트벤치 실행 결과 *reset\_n* 입력값이 변하거나 *j*, *k* 입력값이 변해도 *q*와 *q\_-*가 서로 반대로 잘 나오는 것을 확인할 수 있었다. 마지막으로, *j*, *k*값이 동시에 변하는 것을 다양한 상황에서 실험해본 결과 Master-Slave JK 플립플롭이 글리치를 완화하는 역할을 잘 수행하고 있는 것 역시 확인할 수 있었다.

iii) lab5\_fpga.v

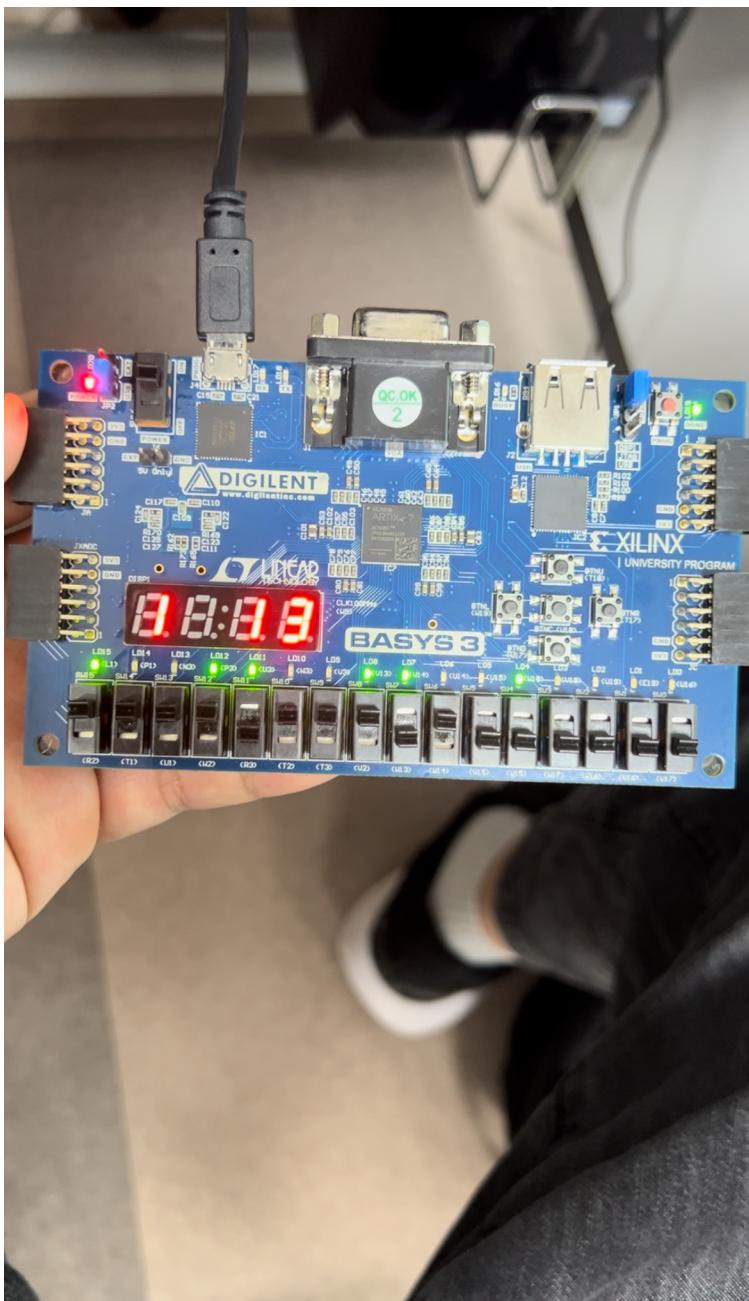
1. arg0=10, arg1=5, op=8



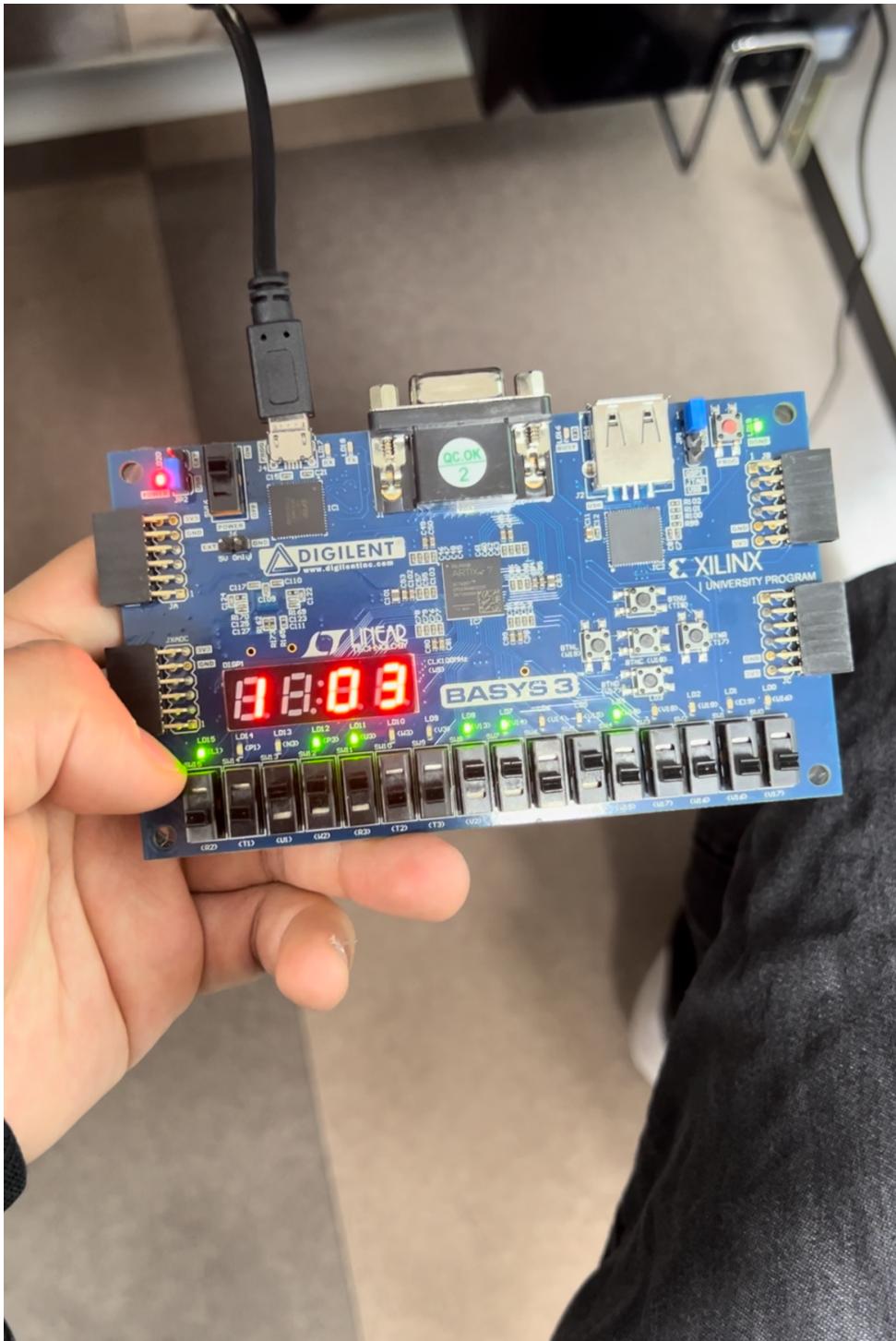
2. arg0=12, arg1=10, op=10



3. arg0=15, arg1=14, op=2



4. arg0=10, arg1=5, op=10



lab5\_fpga.v 파일을 이용해 ALU를 FPGA 회로에 구현한 결과 각 연산이 잘 수행된 것을 확인할 수 있었다.

## 5. 논의

Arithmetic Unit, Logic Unit을 각각 설계하고 이를 합쳐 ALU를 verilog로 설계했다. 그리고 FPGA를 이용해 실제 회로로 올리고 실험함으로써 CPU의 가장 기본적인 단위를 직접 사용하고 경험할 수 있었다. Testbench에서 계속 Wrong이 발생해 디버깅하는데 어려움이 있었지만 오류를 추적하고 문제를 해결함으로써 문제해결력을 기를 수 있었다. 또한, SR 래치를 이용해 master-slave latch를 설계하고 testbench를 이용해 실험했고 잘 작동함을 확인할 수 있었다.