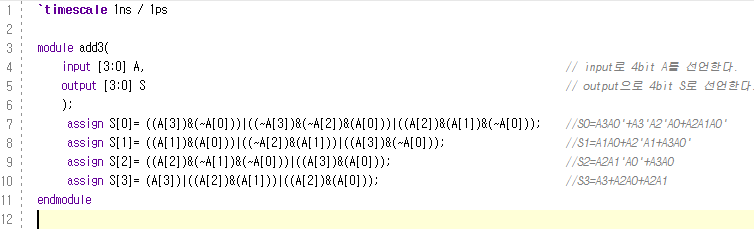
**Lab1.**

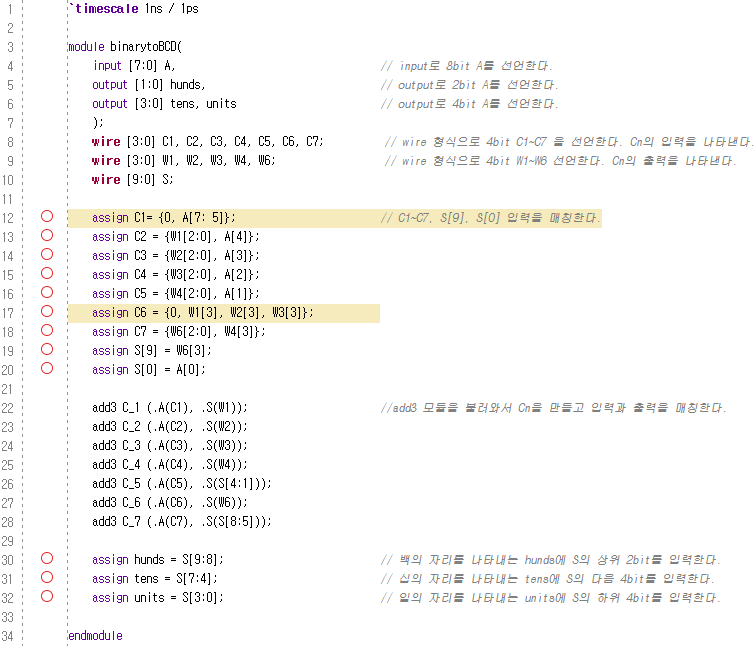
# **Homework1. 8-bit binary to bcd converter**

**Verilog Code / 주석**

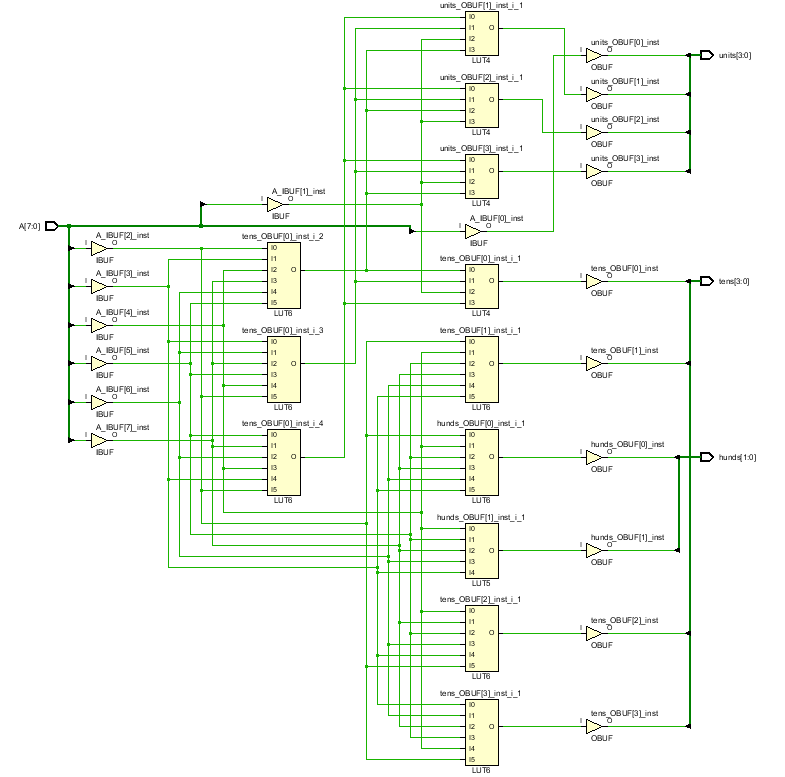
Add3.v



binarytoBCD



**Synthesis Report**

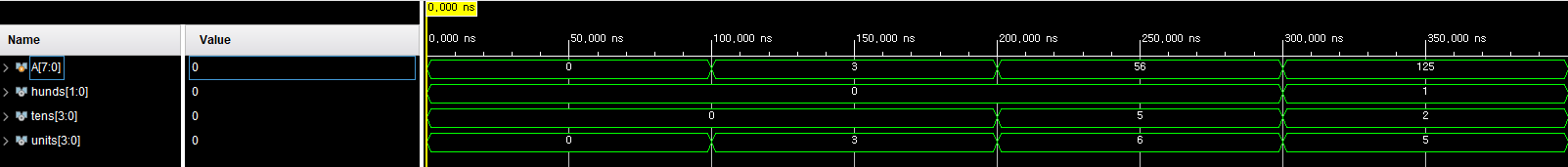
****

**Test Bench Code**

테이블이(가) 표시된 사진

자동 생성된 설명

**Simulation Result**



|  |
| --- |
| **Discussion** |
| . - Verilog Coding을 시작하기 전 작성한 Block Diagram    -Add-3 Module -Binary to BCD Converter(8bit)  - 작성한 Verilog Module에 대한 설명  이번 실습에서는 Add3Module 하나를 Dataflow level로 설계하였다.  8bit의 입력을 A로 받아서 계산을 진행하였다.  C1~C7을 이용하여 binary code를 BCD로 변경시켰다.    각 C는 위와 같이 동작한다.  3bit의 입력을 받고 범위에 따라 값을 더하거나 x로 변경한다.  이 C에는 10이상의 수가 들어올 수 없다.  따라서 10이상의 수가 들어오면 x로 표현하여 오류를 검출할 수 있도록 하였다.    Truth table을 이용하여 S에대해 표현하면 위 그림과 같다.  이 방법을 사용하면 경우에 따라 3을 더하고 shift하는 기능을 구현할 수 있다.  이 과정을 따라 모든 C를 거치게 되면 10bit의 S가 출력된다.  이 값을 비트로 나누어 백의자리, 십의자리, 일의자리가 나오도록 하였다.  - 작성한 Code가 어떤 동작을 하는지? 왜 그런 동작을 하는지?  Code는 먼저 binarytoBCD에서 add3를 불러오는 방식으로 구현하였다.  input과 output, 중간에 wire를 설정하여 중간 값을 받을 수 있도록 설정하였다.  이후 Cn에 대한 입력을 그림과 동일하도록 작성하고 add3 모듈을 불러와서 입력을 입력하고 출력을 불러와 S를 완성하였다.  이후 hunds와 tens, units로 S를 구분하여 입력하였다.  - 작성한 Test Bench Code가 어떤 동작을 하는지?  Testbench는 binarytoBCD를 불러오고 입력을 연결하고 입력해주었다.  0, 3, 56, 125를 100s의 delay의 간격으로 입력하여 표현하였고 이에 따른 출력을 확인하였다.  - Simulation 파형이 왜 그렇게 나온것인지?  Simulation 결과 백의자리, 십의자리, 일의자리가 정상적으로 표현되었다.  - 작성한 Code가 잘 동작하지 않는다면 어디서 문제가 발생한 것인지? 문제없이 잘 동작하였다.  - 오류를 해결했다면, 어떤 부분에서 해결을 했고, 왜 오류가 발생하였는지? 문제없이 잘 동작하였다. |