
# Assignment5

## Simulation Process

| | |
|---|---|
| 과목명 | 하드웨어소프트웨어 통합설계 |
| 담당교수 | 이준환 교수님 |
| 학과 | 컴퓨터정보공학부 |
| 학년 | 3학년 |
| 학번 | 2019202009 |
| 이름 | 서여지 |
| 제출일 | 2021.11.13 (토) |

KWANGWOON UNIVERSITY

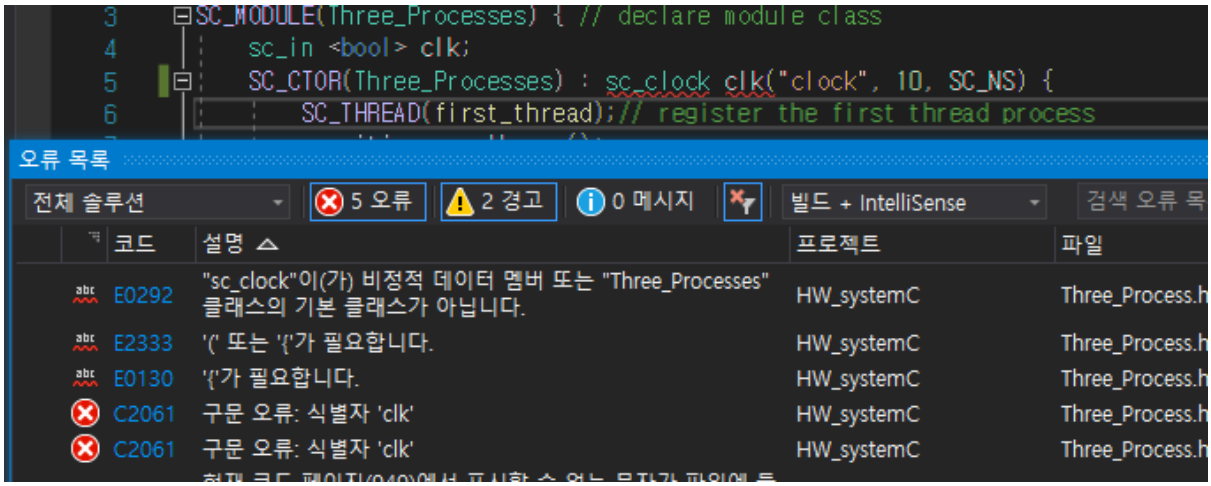
1. 과제개요

여러 개의 thread가 존재할 때 simulation 과정을 확인하는 실습을 진행한다. 세 개의 thread를 갖는 모듈의 동작을 확인하는 예제를 실행한다. 모듈의 각 thread는 무한반복하는 while반복문에서 wait()한 후 몇 번째 thread인지 출력한다.

2. 코드설명

Three_Process.h 파일에는 Three_Processes 모듈과 모듈의 생성자가 정의되어있고, first_thread, second_thread, third_thread가 선언되어있다. 모듈의 생성자는 SC_THREAD를 이용하여 각 thread를 동작하게 하고, sensitive list에 clock의 positive edge를 추가한다. Three_Process.cpp 파일에는 first_thread, second_thread, third_thread의 동작이 정의되어있다. 각 thread는 무한 반복하는 while반복문에서 wait한 후, 몇 번째 thread인지 출력한다. 마지막으로 main5.cpp파일의 sc_main함수에서는 Three_Processes의 인스턴스를 만들고, sc_start를 이용하여 simulation을 실행한다. 이후 Three_Process.cpp의 내용에 wait(SC_ZERO_TIME)을 추가하여 결과를 확인한다.

3. 실행결과




모듈의 생성자에서 sc_clock을 이용할 때 문제가 발생하였고, 이를 해결하지 못하여 예제를 직접 실행시키지 못하였다.

실습자료에 포함된 simulation 결과는 다음과 같이 해석할 수 있다. 가장 먼저 simulation이 실행

된 이후에 clock으로 지정한 10ns마다 세 개의 thread가 출력문을 출력하고 wait한다. 이때 각 thread가 동작하는 순서는 정해져있지 않기 때문에 first-second-third의 순서로 실행되지 않았다. 그러나 wait()이후 wait(SC_ZERO_TIME)을 second_thread와 third_thread에 각각 한 번과 두 번 추가하여 실행했을 때의 결과는 항상 first-second-third의 순서로 실행됨을 확인할 수 있다. ZERO-TIME을 기다리지 않는 thread의 계산 결과가 update된 이후에 ZERO-TIME을 기다린 thread가 동작한다.