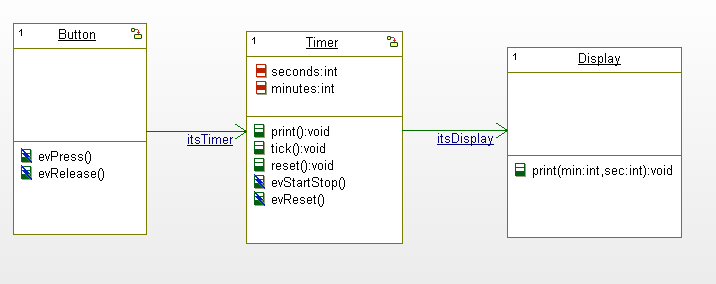
|  |  |
| --- | --- |
| 년도-학기 | 2021년 1학기 |
| 과목명 | 임베디드시스템설계 |

|  |  |
| --- | --- |
| **LAB번호** | **제목** |
| 8 | RiCpp\_RPi.pdf의 Exercise 3: Stopwatch 추가 제출 |

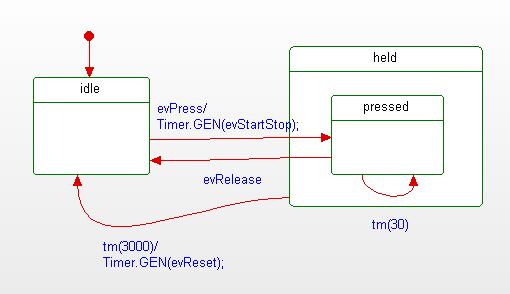
|  |  |
| --- | --- |
| 실험 일자 | 2021년 4월 30일 |
| 제출자 이름 | 강\*\* |
| 제출자 학번 | 201803\*\*\*\* |
| 팀원 이름 |  |
|  |  |

**Chapter 1. 프로그램의 동작 방식 설명**

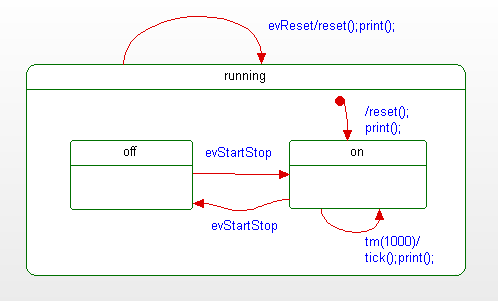
프로그램의 동작 방식을 설명할 때 object model diagram을 참고하면 좋을 것 같아 사진을 첨부했다. 프로그램의 전체적인 구조를 파악할 수 있다.



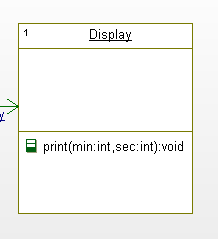
Button object는 다음과 같은 statechart를 가진다. 처음엔 idle 상태에서 기다린다. evPress가 발생하면 Timer에 evStartStop을 발생시키고 pressed state에서 30ms마다 버튼의 입력 값을 확인한다. 버튼 입력이 감지되면 evRelease를 발생시킨다. 이는 polling 방식에서 버튼 채터링을 방지하기 위함이다.



Timer object에서는 시간을 측정하고 출력한다. 처음 시작하면 콘솔 창에는 0:0이 출력되고 1초마다 증가된 시간을 출력한다. evStartStop이 발생하면 시간이 멈춘다. 다시 evStartStop이 발생하면 다시 on state에서 시간을 계속 측정한다. 버튼이 또 눌리게 되면 evStartStop이 발생하고, off state로 가서 멈춘다. 이 과정을 반복하며 시계의 on/off를 반복한다.

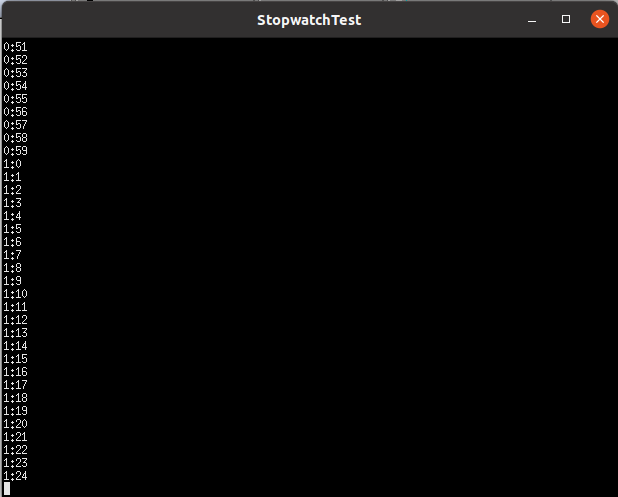


Display object는 Timer의 tick 함수에서 측정한 시간(minutes, seconds)을 받아 콘솔창에 출력하는 함수 print를 가진다.

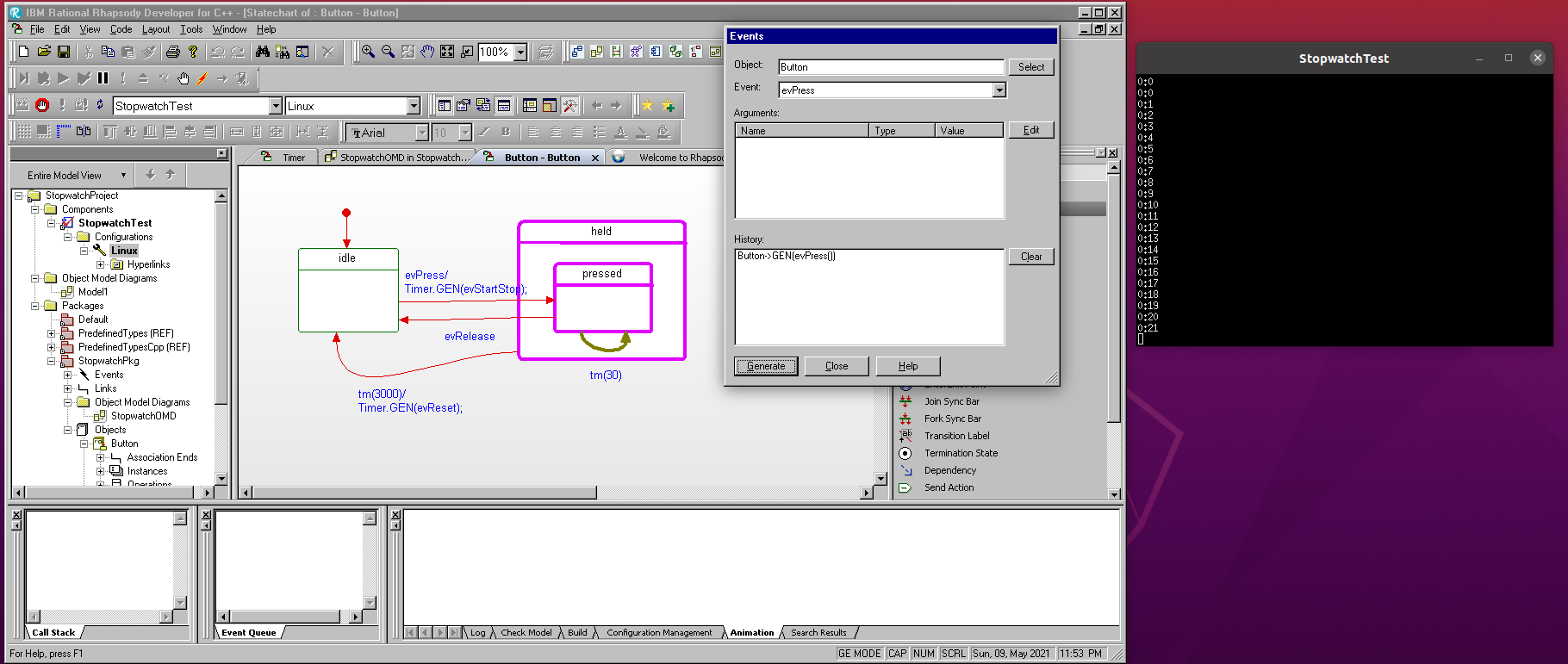


위의 기능들을 수행하면서 stopwatch로 동작한다.

**Chapter 2. 결과**



시간이 잘 출력되고 있음을 볼 수 있다.



Button을 누르면 멈추는 것도 확인할 수 있다.

**Chapter 3. 결론 및 Discussion**

이전 lab 8과제도 잘 되었지만, 이미 예제를 연습하면서 실행해본 예제라 추가 과제 보고서를 작성했다. 시간이 잘 흐르는 것을 볼 수 있었고, button Generate를 통해 버튼이 눌리면 다른 state로 이동하는 것도 확인할 수 있었다. 강의자료를 따라 해본 것이기 때문에 큰 어려움없이 해볼 수 있었다.

**부록**

프로그램 폴더 별도 첨부