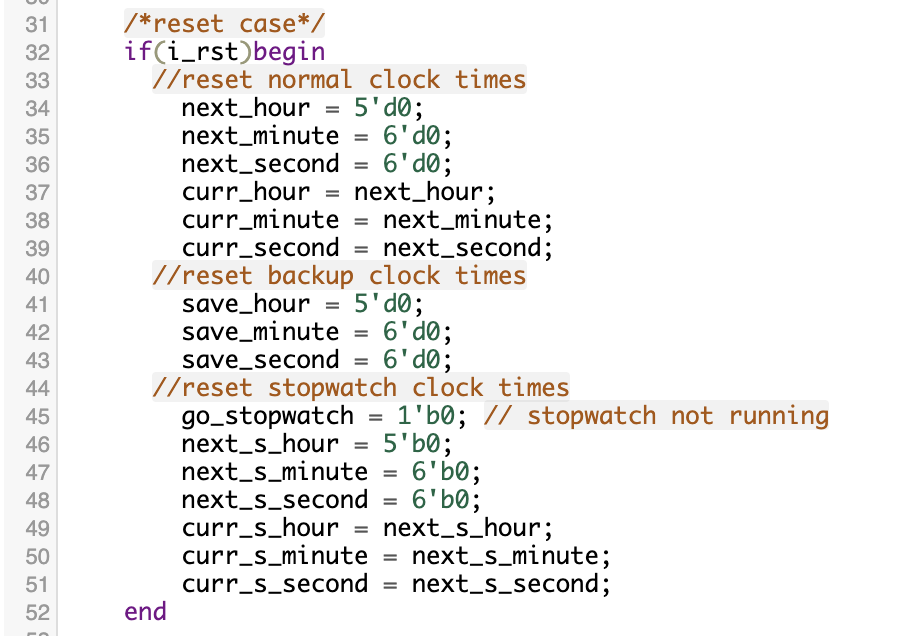
디지털시스템 EE303(B) project report

20190452 Lee Mingyu

1. Brief information
   1. This is a System Verilog code for a EE303(B) project to make a Digital watch. The coding was proceeded in EDA playground. The module ‘digital\_watch’ is a building block for the digital watch. It’s template was presented from the lecture. The inputs for the module are input wire i\_clk, input wire i\_rst, input wire i\_adjust\_time, input wire [4:0] i\_hour\_new, input wire [5:0] i\_minute\_new, input wire [5:0] i\_second\_new, input wire i\_stopwatch\_mode, input wire i\_stopwatch\_start, input wire i\_stopwatch\_stop. The outputs are output logic [4:0] o\_hour, output logic [5:0] o\_minute, output logic [5:0] o\_second.
   2. There are 2 blocks in the code, always\_ff, always\_comb. always\_ff is used for updates, and always\_comb for process.
   3. All of the behaviors implemented by the code are based on the waveforms given in Lecture 7 and the answers given by the TA in the Classum.
2. Workflow
   1. I coded in the order of the inputs and outputs in the design interface given in Lecture 7.
   2. For my implementation, I first implemented a reset, followed by the ability to increment the seconds, minutes, and hours based on a time signal. And I coded it so that the seconds would be 59, the minutes would be 59, and the hours would go back to 0 the moment they went over 23.
      1. When reset is set to 1, time must be set to 0 instant, but has to flow after a slight dely. (asynchronous positive reset).
   3. Then for adjust, I made the new time signals to go to the output times. Using the delay, I made it adjust occur on the next positive edge clock signals.
   4. For final, the stop-watch implementation, I made new timeline to go with the current time (save\_time). Also, I made a variable to tell if the stopwatch is going or not, determined by start, stop signals(go\_stopwatch). And when on stopwatch mode, the output time signals should be from stopwatch timeline.
      1. But the output time should be set to stopwatch and dismissed after 1 clock cycle delay. And stopwatch time should flow 1 clock cycle after start signal is 1, but stop exactly when stop signal is 1.
3. Normal Mode – Reset
   1. 텍스트, 스크린샷, 폰트이(가) 표시된 사진

      자동 생성된 설명
   2. Normal Clock: under codes are valid when clock signal is 1.
      1. 139~141 : the current time is updated to next time.
         1. When right after reset, next time increments but current gets next time before increment. So, even the reset is 0, the current time is still 0 for 1 clock cycle.
         2. Else normal clock times, current time gets the incremented time.
      2. 143~145 : the next time is loaded from the backup time. Not for normal clock actually.
         1. When right after reset, they are all 0.
         2. Else normal clock times, next time and save time is still same.
      3. 147~164: the time signals are incremented. If the exceed the time conditions, they are set back to 0.
      4. 166~168: the next time is saved to save time. Not for normal clock actually.
   3. Reset
      1. 34~51 : all time lines are set to 0. Current and next times are all set to 0 for insurance.
4. Normal Mode – Adjust
   1. 텍스트, 스크린샷, 폰트이(가) 표시된 사진

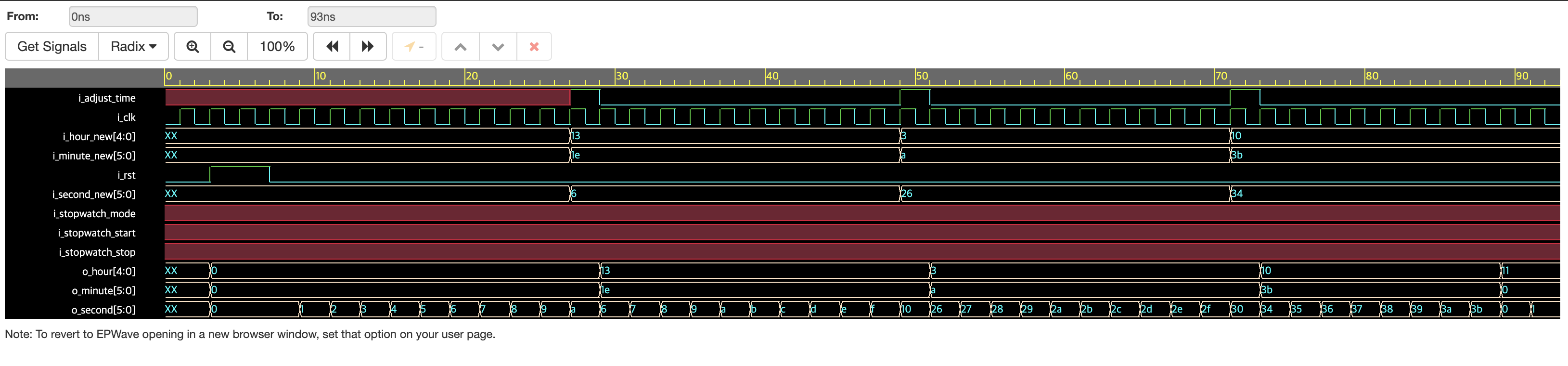
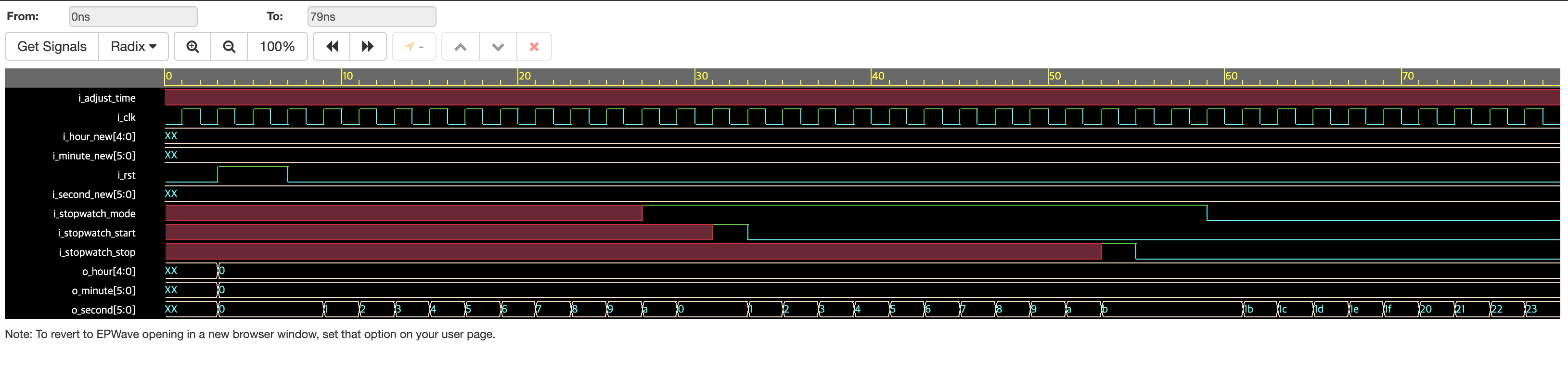
      자동 생성된 설명텍스트, 스크린샷, 폰트이(가) 표시된 사진

      자동 생성된 설명
   2. 58~64 : when adjust time signal is 1, the current time is updated to next time signals. Also, the next time signals are updated to new time signals.
   3. 66~68 : also the backup time is updated to next times. So they are still the same.
   4. After adjust signal is back to 0 in 1clock cycle, we go back to normal clock case.
      1. when start, the current time gets updated to next time. Which is a new time. The next time is loaded but it is same, so its’ not used here.
      2. The time keeps incrementing afterward and update it to save time, so they can follow.
5. Stop-watch Mode
   1. 텍스트, 문서, 스크린샷, 메뉴이(가) 표시된 사진

      자동 생성된 설명텍스트, 스크린샷, 폰트이(가) 표시된 사진

      자동 생성된 설명텍스트, 스크린샷, 폰트이(가) 표시된 사진

      자동 생성된 설명
   2. 73: under codes are valid when clock signal is 1.
   3. 77~79 : current time is updated to next time.
      1. time still flows right when stopwatch mode is 1. So the current time gets next incremented normal time. But after in stopwatch mode, current time get next time, which is updated from next\_s\_time in line 131~133, which is yet 0 but will increment when stopwatch is running.
      2. Right when stopwatch mode is 0, the next time gets updated from save time, so current time is updated to next time(which is just normal time) after 1 clock cycle.
   4. 81~98: Even though in stopwatch mode, the backup time should be running to prepare for exit of stopwatch mode.
   5. 100~103: current stopwatch time is updated to next stopwatch time
      1. When stopwatch is not going, they are both 0.
      2. But when stopwatch is going, current stopwatch time gets next stopwatch time. next\_s\_time increase when stopwatch is going so current time increases after 1 clock cycle.
      3. When stopwatch stops, next stopwatch time is constant, but current stopwatch gets 1 increased time at the moment but doesn’t increase afterwards.
   6. 104~127 : the stopwatch time increments when stopwatch is running.
      1. When start signal, go\_stopwatch is 1, next\_s\_time starts increment, so next time also increment. As a result current time, output time increments after 1 clock cycle.
      2. When stop signal, go\_stopwatch is 0, so next\_s\_time becomes constant, current time, output time increments this time, becomes constant after 1 clock cycle.
   7. 131~134: the next time is updated to next stopwatch time, so that current time can be updated with delay and presented to output.
6. Waveforms about three scenarios / EPWave result
   1. Reset
      1. 스크린샷, 텍스트, 라인이(가) 표시된 사진

         자동 생성된 설명
   2. Adjust
      1. 
   3. Stop-watch
      1. 
   4. Custom made
      1. 스크린샷, 멀티미디어 소프트웨어, 디자인이(가) 표시된 사진

         자동 생성된 설명