Lab Class 6
09.11.2021

- **Waveform generation.**

Problems

1. Implementation of a sawtooth waveform generator:
   - implement a simple sawtooth waveform generator by connecting a counter to a DAC;
   - by using the leftmost switch, implement the option that allows to change the sign of the waveform slope;
   - by using the other switches, implement an option that allows to change the absolute value of the waveform slope, and thus of the frequency.

At the end, show the result to the lecturer.

Upon eliminating the unuseful files (only *.v*, *.ucf*, *.xise* are necessary), compress each working folder via
`tar czf labClass_6_<names>.tgz <Folder>`
and upload the resulting compressed file to the Moodle platform.

Additional problems (to solve in the lab or as homework)

- Implementation of a triangular waveform generator:
  - implement a counter that inverts its "counting direction" (up/down) whenever an upper limit (it can be $2^{12} - 1$) as well as a lower limit (it can be 0) is reached;
  - implement a triangular waveform generator by connecting the new counter to a DAC;
  - by using the switches, implement an option that allows to change the waveform frequency.
- Implementation of an 8–bit shift register:
  - implement an 8–bit shift register (SR) and display its function by
    - clocking the SR with a 1 or 2 Hz source,
    - providing the SR's serial input via a pushbutton, and
    - connecting the flip–flop outputs to the eight LEDs;
  - by using a switch (alternatively, a pusbutton driving a toggle flip–flop can be used) implement the "closed loop" option.
- Implementation of a 31–bit pseudorandom number generator:
  - implement a pseudorandom number generator and display its function by
    - implementing a 31–bit shift register (SR),
    - feeding–back the serial input (i.e. the 0^{th} bit) with the XOR of the 30^{th} and the 27^{th} bit,
    - clocking the SR with a 1 or 2 Hz source and
    - connecting eight *randomly selected* flip–flop outputs to the eight LEDs;
- Implementation of a 31–bit pseudorandom noise generator (oscilloscope required!):
  - implement a pseudorandom noise generator by
    - clocking the previous circuit at a frequency of ~ 1 kHz or higher,
    - generating a 12–bit pseudorandom number by concatenating the outputs of 12 randomly chosen flip–flops,
    - feeding this number to a DAC (if an oscilloscope is available).

Upon eliminating the unuseful files (only *.v*, *.ucf*, *.xise* are necessary), compress each working folder via
`tar czf labClass_6_<names>_additional.tgz <Folder>`
and upload the resulting compressed file to the Moodle platform.

At the first favorable circumstance, show the result to the lecturer.