⚠ You have previously submitted this assignment with David Rama Jimeno. Group can only change between different assignments.

This course has already ended.

« 9. Exercise 6 (/comp.ce.240/spring-2024/... 10. Exercise 7 » (/comp.ce.240/spring-2024...

COMP.CE.240 (/comp.ce.240/spring-2024/) / 9. Exercise 6 (/comp.ce.240/spring-2024/E06/) / 9.1 Exercise 06: Triangular wave generator

Assignment description (/comp.ce.240/spring-2024/E06/wave_gen/)

My submissions (2/1000)

Exercise 06: Triangular wave generator

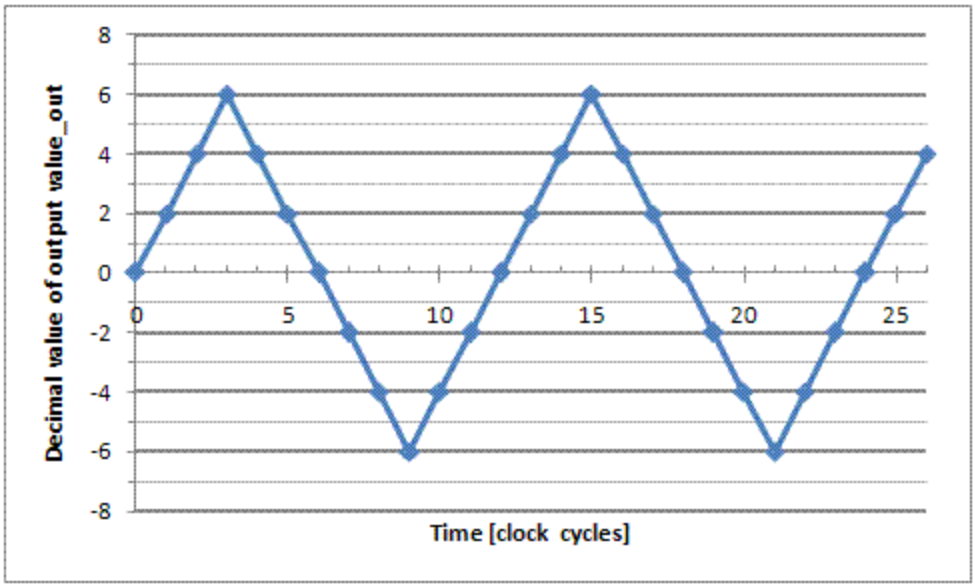
The purpose of the exercise is to learn:

- the implementation of counters and synchronous logic
- the usage of the waveform window in ModelSim

Your task is to implement and verify a parameterizable triangular wave generator

- The implementation is bidirectional step counter which uses signed values
- The generator counts from the smallest to the largest value and vice versa using predefined steps
- After reset, the counter starts from value zero and counting begins **upwards**
- The behaviour of the counter is illustrated in the figure below
  - In the example, the counter width b=4 bits and step s=2
  - 4-bit wide value range is -8..7. When the step is 2, count takes place on the range -6..6.

- The generator can be shut off by setting signal *sync_clear_n* to '0' (note: active low signal)
  - This signal keeps output as zero and sets the count direction upwards. When *sync_clear_n* is set again to '1', the counting begins normally from zero upwards.
  - The behaviour is similar to `rst_n` but synchronous. Remember not to add extra logic to `rst_n`-signal nor this signal to a synchronous process's sensitivity list!



- The count direction changes when either of the limit values is reached
- Maximum and minimum limits are set with the following expression

$$\max = \left\lfloor \frac{2^{b-1} - 1}{s} \right\rfloor s$$

$$\min = -\max$$

  - b is counter width as bits
  - s is step size
  - both b and s are positive integers (in the example figure b=4 and s=2)

Creating the wave generator

- Specification of the block interface:
  - Entity name `wave_gen`
  - Generic parameters (Generics)
    - `width_g` (b in previous expression)
    - `step_g` (s in previous expression)
  - Inputs
    - `clk`
    - `rst_n`
    - `sync_clear_n_in`, 1 bit
  - Output
    - `value_out` with width `width_g` (two's complement format)

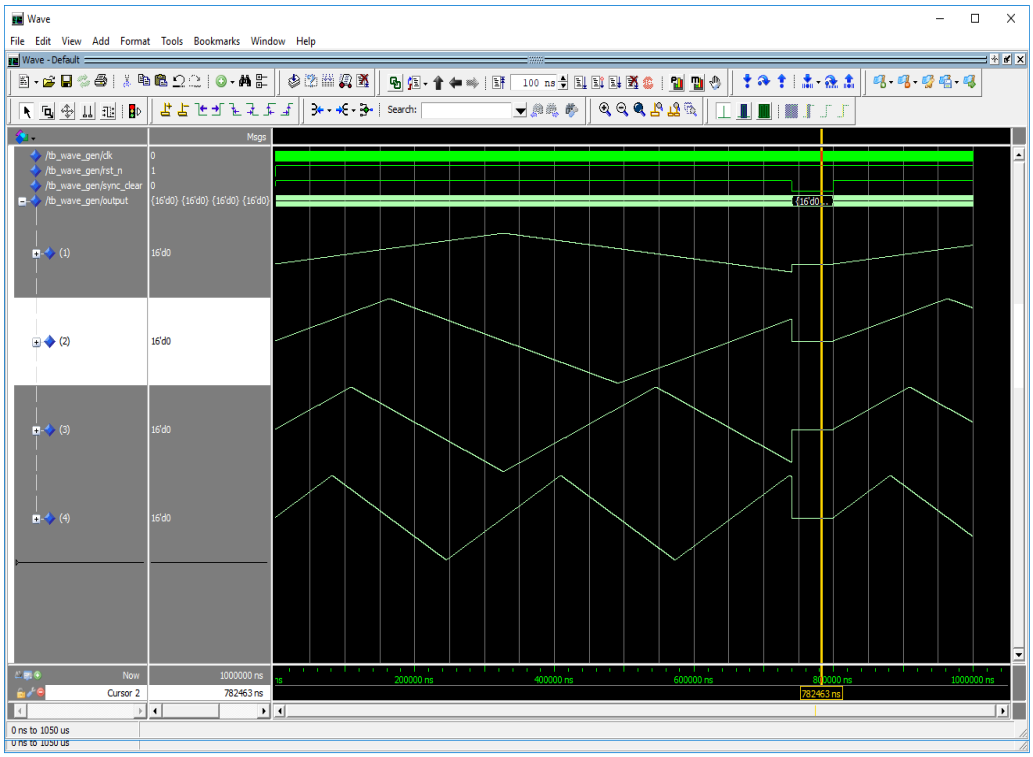
- Define constants, signals and processes, and implement the desired functionality
  - General instructions and hints
    - Power operator is `**` (e.g., `2**width_g`)
    - Avoid using variables as instructed in the course's general coding guidelines
    - Prefer precise comparison (=) instead of relative (>, <, >=, <=)

        - In digital logic, it is straightforward to check equality and inequality (= and /=) using XOR-ports but relative operations (<, >) require comparator unit whose area is larger and it's pass-through delay is longer.

- Compile your code using command **vcom** and simulate with command **vsim** using the given test bench (https://plus.tuni.fi/graderA/static/compce240-f2021/E06/tb_wave_gen.vhd) for 1 ms.
    - Add test bench signals to waveform. Input generator initializes four wave generators with different steps. Array `output` contains four signals. The array's index depicts corresponding step size.
    - Set the array's `output` signals according to the following instructions:
        - Set `output`-array's signals to state "analog (custom)"
        - Use a slightly larger range than 16 bits, e.g., -32800...32800
        - Check "Clamp waveform within row" to get a finer waveform
        - Choose properties for signal `output` and change the signal height to value 100
        - Set radix to decimal
        - Use the dark blue magnifying glass for signal scaling
        - The resulting scaled waveform should look similar to the following figure

- Signals change relatively slowly and values are large with a 16b generator. Try to change the widths smaller from the test bench, e.g., 6b. In this way, it can also be verfied that generic values are used correctly.

- **Important attention!** This "test bench" generates only input and it does not make any testing of the output's correctness. Testing is made manually by looking at the waveforms which is not preferable in real designs! However, this exercise focuses on showing the features of the analog view mode in ModelSim, so it is an exception made for educational purposes :)



(https://plus.tuni.fi/graderA/static/compce240-f2021/E06/reference_waveforms_new.png)

FAQ

- How to implement the floor function? Can I use the floor function in VHDL's math package?
  - The floor function in VHDL's math package is only for *real* type values so it is not synthesizable. Here we can exploit the rounding rules of integers in division (always downwards).
- Analog representation jumps from maximum to minimum or behaves oddly
  - Check that radix decimal (=signed) is in use and that correct checks for maximum and minimum limits are used in the code. Remember that assignment to a signal in a process incurs one cycle delay!

BONUS (https://plus.tuni.fi/comp.ce.240/spring-2024/bonus/sine_wave_gen/): A wave generator version which produces a sine wave instead of a triangular wave.

Return:

- Put your returned files under `E06` folder in your Git repository
  - Return your own `wave_gen.vhd`
- Check that the files' header comments are valid, made according to instructions, and you have followed the coding rules
- Push the changes to your repository and submit (with your partner if you have a group!)
  - Use the **ssh variant** of the repository url in the submission. Otherwise the tests will fail 100% even with working design.

- The url looks like *git@course-gitlab.tuni.fi:compce240-spring2024/<your_group_number>.git*.

Enter your Git repository address for grading

Did you remember git add - git commit - git push?

Submit with David Rama Jimeno Submit

Earned points

6 / 6

Exercise info

Assignment category
VHDL exercises

Your submissions
2 / 1000

Points required to pass
6

Deadline
Sunday, 25 February 2024, 23:59

Late submission deadline
Friday, 31 May 2024, 23:59

Group size
1-2

Total number of submitters
62

« 9. Exercise 6 (/comp.ce.240/spring-2024/... 10. Exercise 7 » (/comp.ce.240/spring-2024...