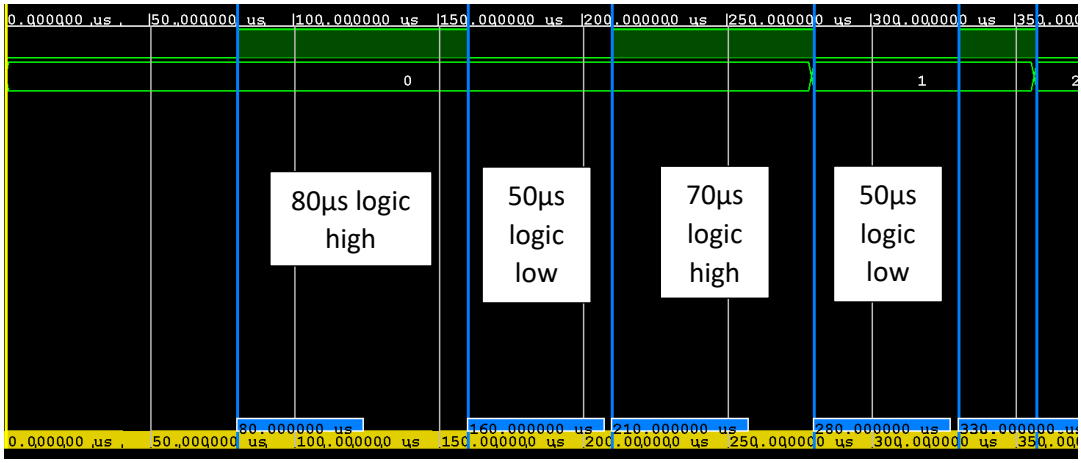
Lab 6: Arbitrary Waveform Generator

Design a VHDL code that generates arbitrary digital waveform(s) of your choosing. You can also make the waveforms change with some input.

- Use “Clocking Wizard IP” to generate any frequency clock you want (within reason). Even if you want to use 100MHz, use Clocking wizard to make a “cleaner” clock.
- You might want to implement any waveform you will need in your project. For example, if you use VGA in your project you need V_{sync} and H_{sync} waveforms. If you use servo motors, you need specific waveforms to control it.
- Write a testbench and simulate your code. Measure the time intervals to make sure the waveform is as intended.
    - If you need logic high for 70µs, make sure it is not 70.01µs (10ns error→1 clock cycle at 100MHz)
    - Use markers in the simulation window to get relative time measurements (As shown in the figure).
- Check 1: Run a simulation and show the test bench results to your TA, also add it in your report.
- Check 2: Output your signal to a pin and measure the generated waveform in oscilloscope.

Notes:

- A sample output is given in the figure below.



- Use only one main clock in your design. This means that there can be only one process and “if rising_edge(s_clk) then” should wrap your entire code. Notice that no other signals are necessary in the sensitivity list other than the clock signal (Except for optional asynchronous reset).

```
process(s_clk)
begin
        if rising_edge(s_clk) then
                --your code here
        end if;
end process;
```

- Although it is possible to use multiple frequencies in one design, if you want to use one signal in both clock domains, you will get a timing error due to clock domain crossing problems.

```
process(s_clk_100MHz)
begin
        if rising_edge(s_clk_100MHz) then
                s_my_signal <= (some operation);
        end if;
end process;
process(s_clk_25MHz)
begin
        if rising_edge(s_clk_25MHz) then
                s_another_signal <= s_my_signal; -- any operation that uses s_my_signal
                -- Clock Domain Crossing, Timing problems!
        end if;
end process;
```

- In your projects we encourage you to use any Xilinx IPs.