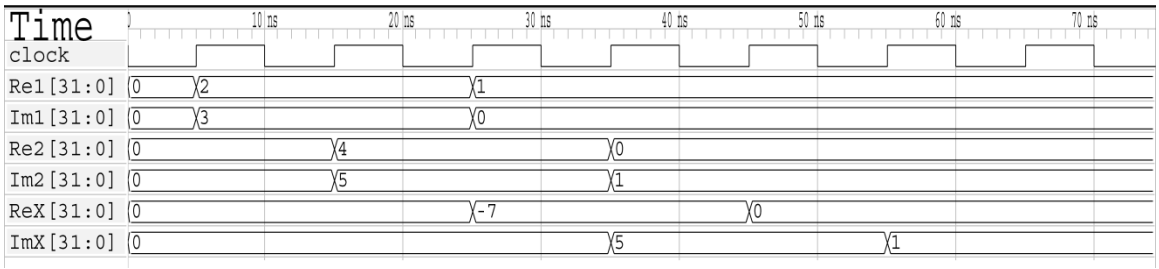
Assignment 1

Due June 6, 13:59

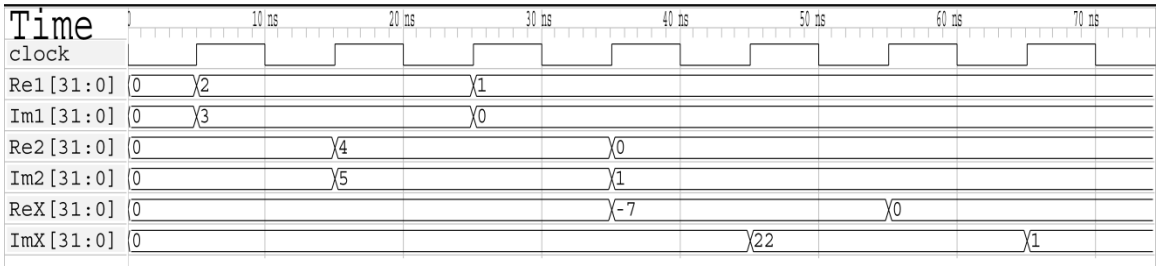
Important: Late submissions will NOT be accepted. Please submit a hardcopy of your solutions in the ECE 466 **drop-box** (ELW, second floor) and your SystemC code via the ECE 466 **CourseSpaces** webpage, following the submission guidelines posted on the course website.

1. *[10 points]* Slides **24** and **25** from the **"SystemC: Part I"** lecture notes show two versions of a 2-cycle complex multiplier. Below are the two corresponding waveform examples (see the course website for the complete code including a testbench). The inputs are the same (**Re1**, **Im1**, **Re2**, **Im2**) in both examples, but the outputs are different (**ReX**, **ImX**). Briefly explain why such outputs have been produced in each case, i.e., explain the observed output values and their timing.

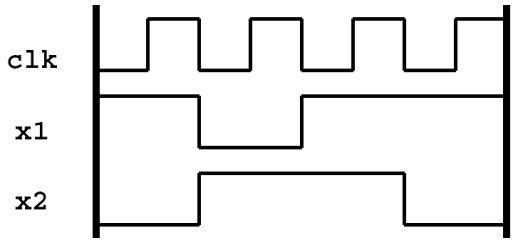
"Bad" Multiplier



"Good" Multiplier



2. [10 points] Consider the SystemC code and the input waveforms of **`clk`**, **`x1`**, and **`x2`** shown below. Draw the corresponding output waveforms of **`y1`**, **`y2`**, and **`y3`**.



```
SC_MODULE (example) {
        sc_in <sc_logic> x1, x2;
        sc_out <sc_logic> y1, y2, y3;
        sc_in_clk clock;

        sc_signal <sc_logic> s;                 // signal "s"


        void example_process1() {
          sc_logic v;                           // variable "v"
          v = ~s.read();                        // ~ means NOT
          y1.write(x1.read() ^ v);              // ^ means XOR
        }

        void example_process2() {
          sc_logic u;                           // variable "u"
          u = ~x2.read();                       // ~ means NOT
          y2.write(s.read() & u);               // & means AND
        }

        void example_process3() {
          while(1) {
             s.write(x1.read() ^ x2.read());    // ^ means XOR
             y3.write(~s.read());               // ~ means NOT
             wait();
          }
        }


        SC_CTOR (example) {

              SC_METHOD(example_process1); sensitive << x1 << s;
              SC_METHOD(example_process2); sensitive << x2 << s;
              SC_CTHREAD(example_process3, clock.pos());

              s.write(SC_LOGIC_0);              // s initially 0
              y1.initialize(SC_LOGIC_0);        // y1 initially 0
              y2.initialize(SC_LOGIC_0);        // y2 initially 0
              y3.initialize(SC_LOGIC_0);        // y3 initially 0

        }

};
```

3. *[20 points]* Using a single module with a single process of type **SC_CTHREAD**, write a SystemC code for the digital filter shown below ($\mathbf{z^{-1}}$ means a delay of 1 clock cycle), whose multiplication coefficients are 0.4, 0.24, –0.8, 0.2, –0.5, and 0.25. In addition to the clock, the filter has one **`float`**-type input port **X** and one **`float`**-type output port **Y**. Your code must include a stimulus generator (e.g., a simple 1-cycle pulse to obtain the impulse response) and a result monitor with waveform tracing.

Please submit a softcopy of your SystemC code via the ECE 466 **CourseSpaces** webpage (following the submission guidelines posted on the course website), and put a hardcopy of your code in the ECE 466 drop-box.