Practical work nº 3 - Programmable timer

1 Goals

Implementation of an algorithm: decomposition into *data path* and *control unit* and generation of the corresponding VHDL model.

2 Specification

The programmable timer has

one 4-bit input *delay*,
four 1-bit inputs *clk*, *reset*, *start* and *reference*,
one 1-bit ouput *done*

(figure 1).

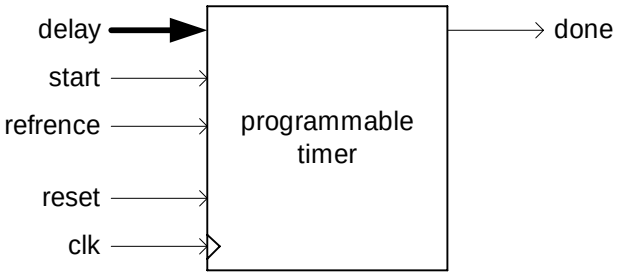

Figure 1 Inputs and output

On every positive (0 to 1) transition of *start*, the circuit reads the value of *delay* and puts *done* at 0. After a time interval roughly equal to the period *T* of the *reference* signal, multiplied by the previously read value of *delay*, *done* goes from 0 to 1 (figure 2)

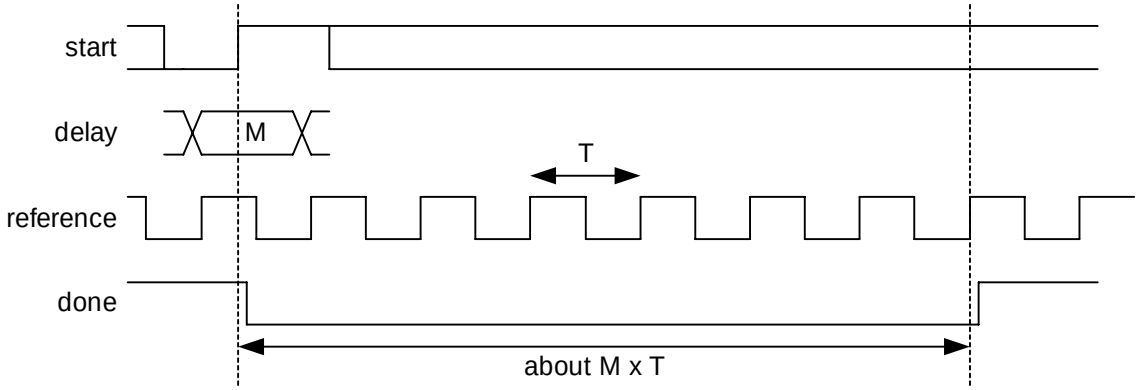

Figure 2 Timer working

The next algorithm describes the timer behavior:

```
while (1 == 1)
{
      while (start == 1); /* wait for start = 0
      while (start == 0); /* wait for a positive edge on start
      x = delay; done = 0; /* start up
      while (x > 0)
      {
            while (reference == 1); /* wait for reference = 0
            while (reference == 0); /* wait for a positive edge on
                                    /*reference
            x = x-1;
      }
      done = 1; /* end
}
```

The circuit is made up of a data path (figure 3) and a control unit (figure 4 and table 1).

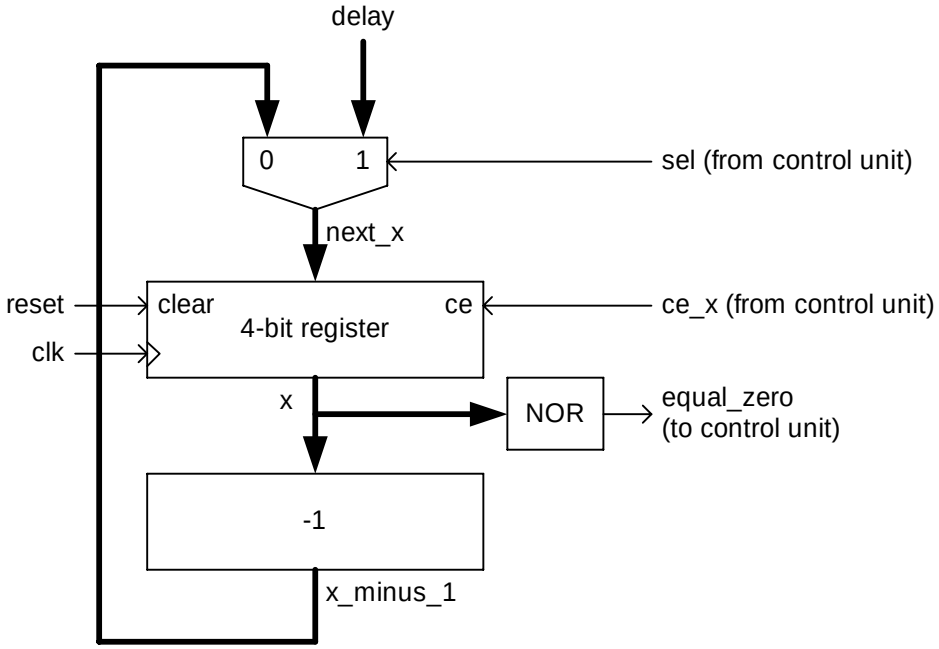

Figure 3 Data path

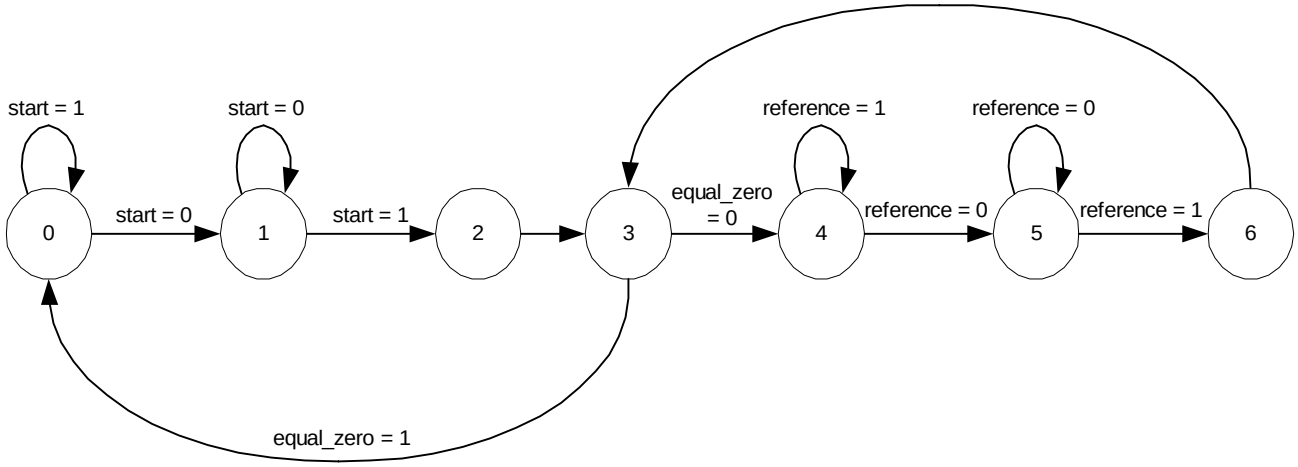

Figure 4 Control unit

state	*operation*	*done*
0	nop	1
1	nop	1
2	x <= delay	0
3	nop	0
4	nop	0
5	nop	0
6	x <= x-1	0

Table 1 Operations and value of *done* in function of the control unit internal state

3 Previous work

3.1 Generate a VHDL model of the data path.

3.2 Generate a command file (.do) for simulating the data path.

3.3 Generate a VHDL model of the control unit.

3.4 Generate a command file for simulating the control unit.

3.5 Generate a VHDL model of the timer. For that purpose, use the data path and the control unit as components (figure 5).

3.6 Generate a command or test bench file for simulating the timer.

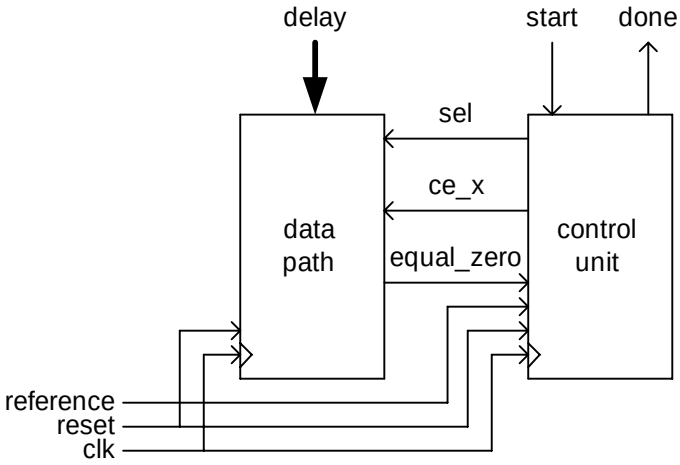

Figure 5 Timer

4 Practical work

4.1 Simulation of the data path, the control unit and the timer.
4.2 (optional) redesign the state machine using STATECad. Simulate.

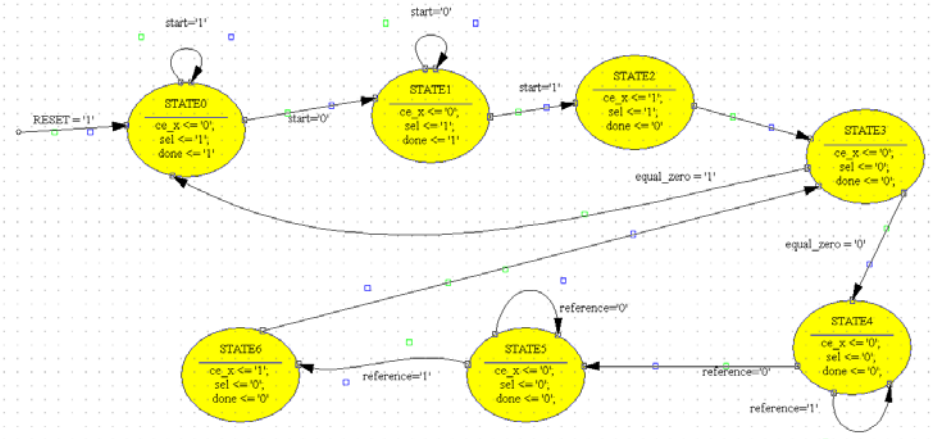

Figure 6 FSM designed with STATECad