CSD laboratory P_Ch2: Programmable timer

SP2_3 Designing the programmable timer control unit

This lab exercise is similar to the problems proposed in P6. Complete specifications of the FSM will be given, thus, planning, developing and testing will follow up as a series of automated design steps using VHDL techniques and CSD design conventions.

NOTE: Solve Lab6 and study some of the tutorial projects in P6 before trying to solve this SP2_3.

1. Specifications

This unit is a component for the programmable timer (chip 1 in P_Ch2 Fig. 3). It advances using external control signals (TRG pulse) and internal status flags from the datapath (TOF flag) accordingly to the state diagram represented in Fig. 2.

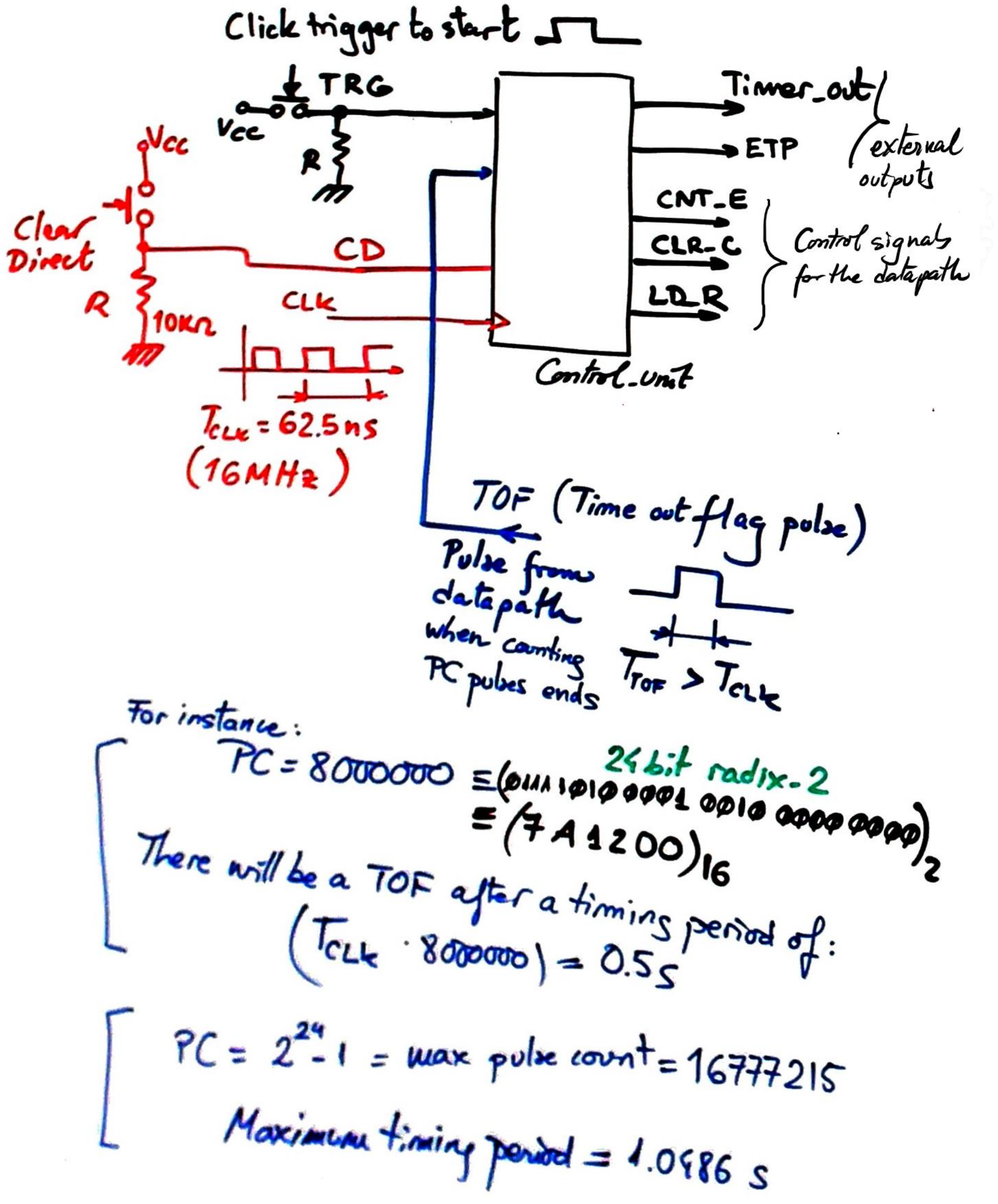

Fig. 1 *Symbol of the programmable timer control unit*

Fig. 2 shows the proposed state diagram for this FSM. It generates at each state the external timer outputs (*Timer_out*, *ETP*) and the internal control signals for running the datapath.

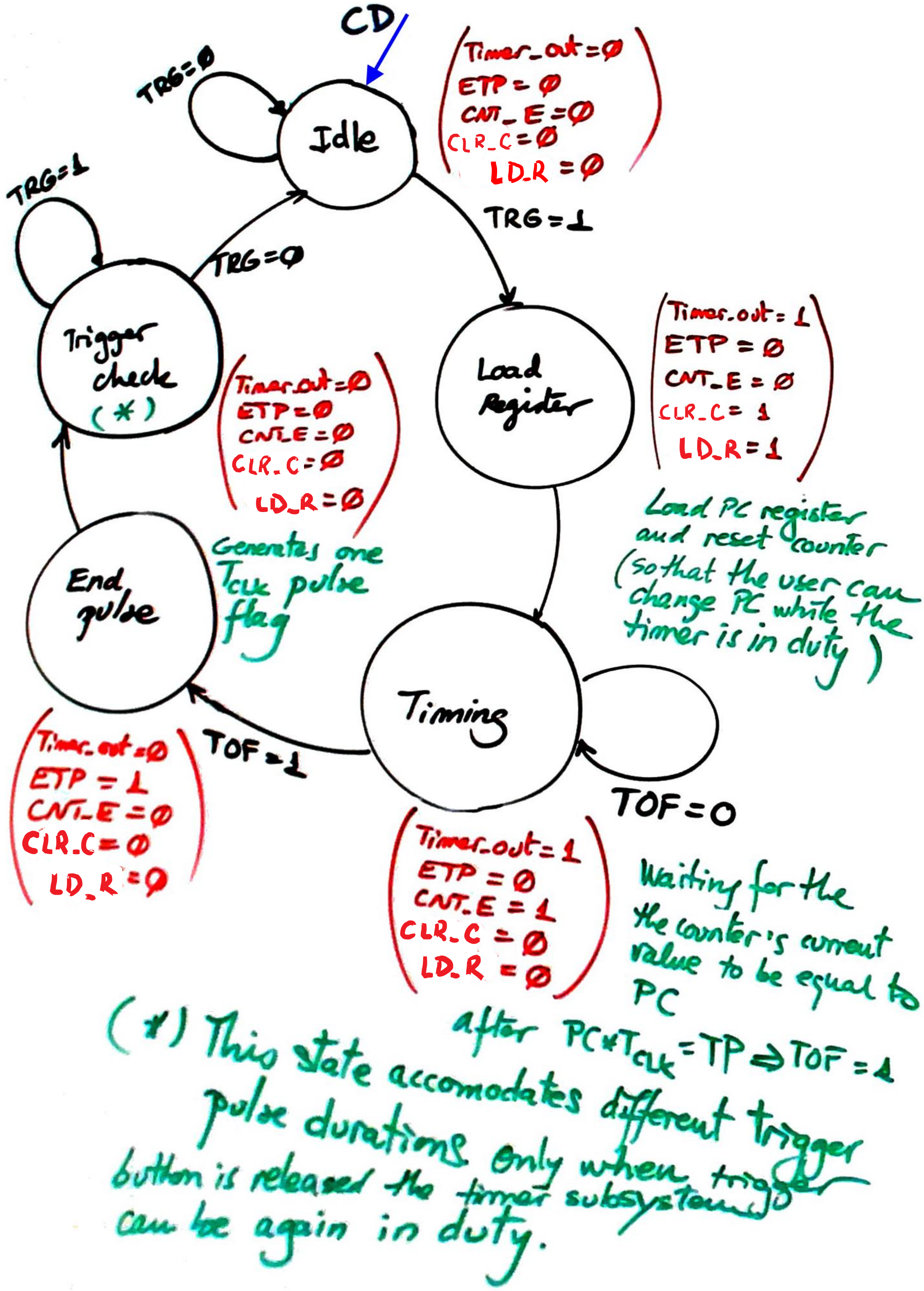

Fig. 2 *State diagram for the control unit. All the black-coloured arrows (state transitions) are sensitive and executed at the CLK's rising edge.*

Fig. 3 shows an example of timing diagram solved for a PC = 200. A timing period of TP = 12.5 µs is generated for a T_{CLK} = 62.5 ns. Using the given parameters, a maximum timing period of about TP = 1 s is attained.

Another interesting feature can be *easily* added to the programmable timer: adjustable time scale, making the device able to time from ns to hours. We can run the datapath from another CLK source independently from the system CLK using the Chip 3 *CLK_Generator* in the dedicated processor architecture.

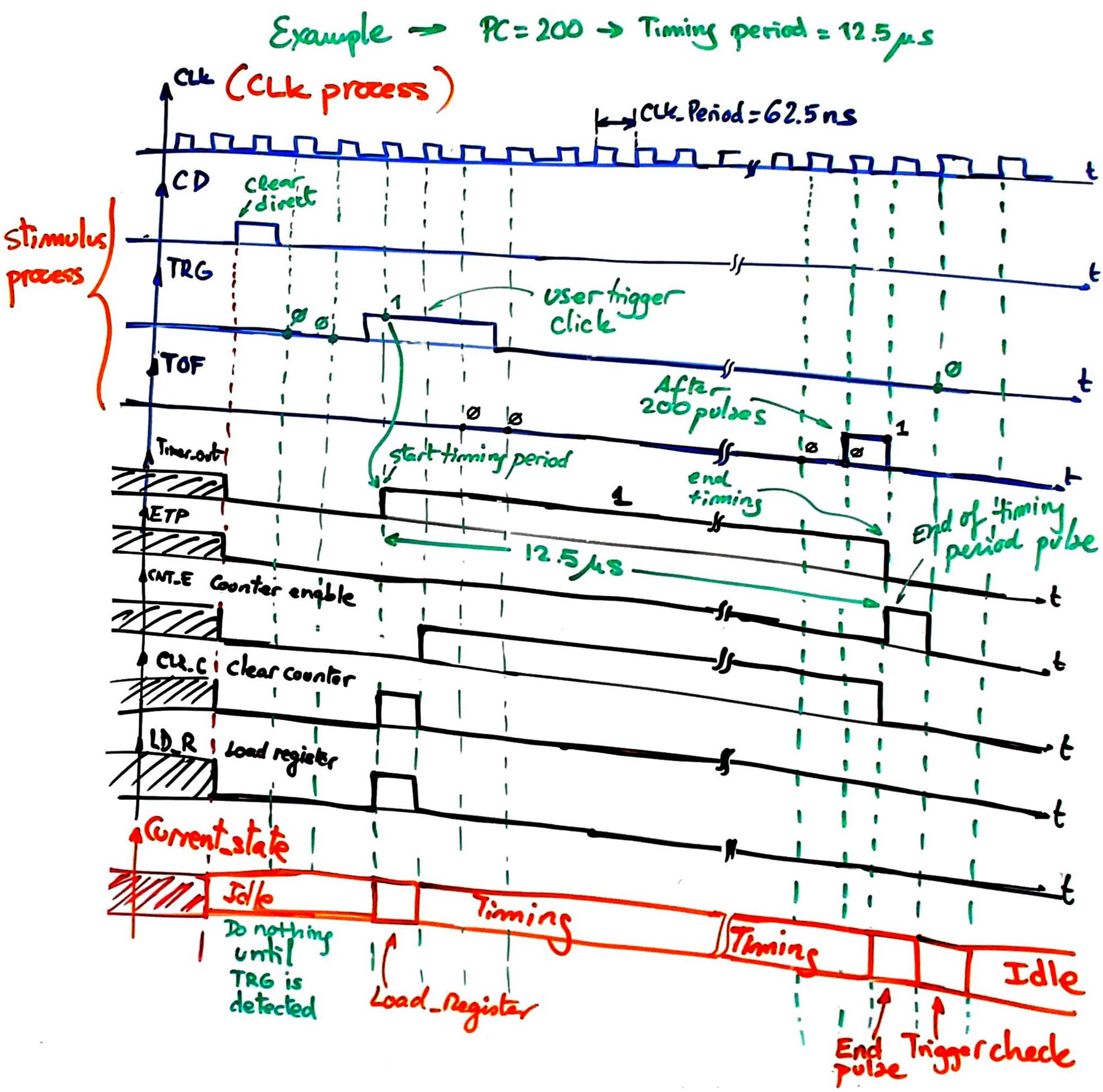

Fig. 3. Example of timing diagram. TRG signal last as long as the user is clicking the trigger button. If the trigger pulse duration is longer than the timing period, the state *Trigger_check* will last for many CLK periods until it goes down, meaning that the user has released trigger button, enabling the machine to perform another timing operation.

2. Planning

Fortunately, from now on everything may be automated. Pen and paper work following our CSD FSM design procedure reproduced here in this design.

a. Draw the general architecture of a finite state machine (FSM).

b. Adapt the general FSM architecture to this problem and draw the state register based on D_FF. Deduce how many D_FF are required if you are coding in binary sequential or in one-hot.

c. Write the truth table of CC1 and its equivalent behavioural interpretation in flowchart.

d. Write the truth table of CC2 and its equivalent behavioural interpretation in flowchart.

3. Developing

EDA synthesis tool.

e. Write the VHDL file *Control_unit.vhd* by translating the flowcharts and the state register. Run a project (*Control_unit_prj*) using an EDA synthesis tool for a CPLD MAXII EPM2210F324C3 or FPGA Cyclone IV EP4CE115F29C7N target chip.

f. Print and discuss the RTL schematic. Count the number of D_FF used in this application.

g. Print and discuss the technology view schematic.

4. Testing

VHDL simulation tools.

Use the given testbench constant `CLK_period`, `CLK_process` and `stim_proc` given in Fig. 4.

```
-- ************************************************************************
-- Clock period definitions
constant CLK_period : time := 62.5 ns; -- 16 MHz
-- ************************************************************************


-- ************************************************************************
-- Clock process definitions
CLK_process :process
begin
CLK <= '0';
wait for 3*CLK_period/5;
CLK <= '1';
wait for 2*CLK_period/5; -- Duty cycle of 40%, rectangular wave
end process;


-- ************************************************************************
-- Stimulus process for signals CD, TRG and TOF
stim_proc: process
begin
wait for CLK_period*7.65;
CD <= '0';
TRG <= '0';
TOF <= '0';
wait for CLK_period*3.53;
-- Clear direct pulse:
CD <= '1';
wait for CLK_period*2.23;
CD <= '0';
wait for CLK_period*6.78;
-- Trigger pulse:
TRG <= '1';
wait for CLK_period*2.23;
TRG <= '0';
-- ************************************************************************
-- From now on the FMS must send automatically control signals to the datapath
-- Once enabled, the datapath counts up at each CLK rising edge.
-- Thus, for instance, waiting now for 200·CLK_period
wait for CLK_period*200;
-- means that the datapath will set the timeout flag (TOF )for the FSM
-- after reaching 200.
-- Let's simulate such datapath response:
TOF <= '1';
wait for CLK_period*1.3;
-- And, so, when sampled, the FSM must generate ETP indicator and be ready at
-- Idle for a new operation.
TOF <= '0';
wait for CLK_period*6.2; -- (or any other time until the next operation)
wait;
end process;
```

Fig. 4. Testbench constant CLK_Period, and CLK and stimulus signals from Fig. 3 translated to VHDL.

h. Start a functional simulation project using a VHDL testbench and discuss the results.

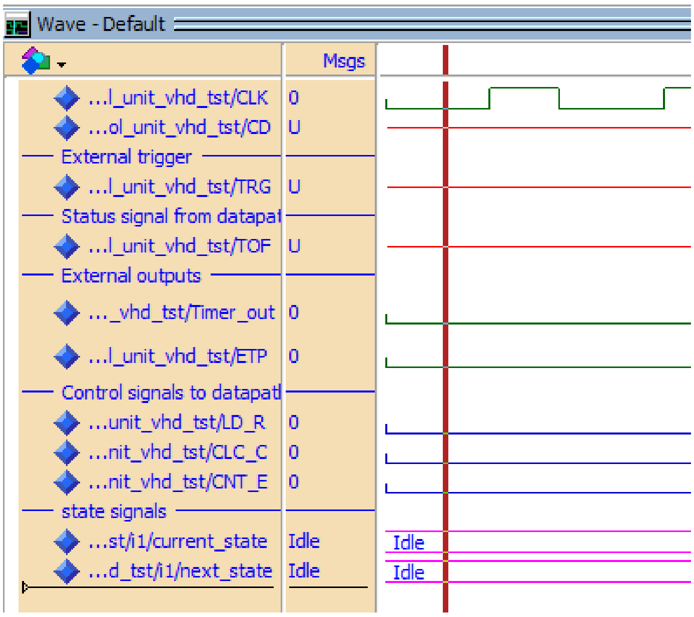

Fig. 5. Example of wave list and dividers to organise result in the logic analyser output.

i. Start a gate-level simulation project and discuss the results. Measure the t_{CO} parameter for the selected target chip.

j. Measure the maximum CLK frequency that can be applied to your design using the timing analyser tool.